TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

TECHNICAL UNIVERSITY OF LIBEREC
Faculty of Mechatronics, Informatics and Interdisciplinary Studies

Vývoj webové aplikace na podporu uspořádání konference

Development of web application to support the organization of a conference

STUDIJNÍ PROGRAM N2612 – ELEKTROTECHNIKA A INFORMATIKA
STUDY PROGRAMME N2612 – ELECTROTECHNOLOGY AND INFORMATICS

STUDIJNÍ OBOR 1802T007 – INFORMAČNÍ TECHNOLOGIE
STUDY BRANCH 1802T007 – INFORMATION TECHNOLOGY

DIPLOMOVÁ PRÁCE | DIPLOMA THESIS

Autor práce | Author — Bc. Jan Červenka
Vedoucí práce | Thesis supervisor — Ing. Igor Kopetschke,Ph.D.

LIBEREC 2013

Tento list nahraďte
originálem zadání.

Prohlášení

Byl jsem seznámen s tím, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé diplomové práce pro vnitřní potřebu TUL.

Užiji-li diplomovou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Diplomovou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím mé diplomové práce a konzultantem.

Datum:

Podpis:

Abstrakt

Tato diplomová práce pojednává o nástrojích, které slouží jako podpora při pořádání konference. Je rozdělena na část analytickou a na část implementační. Do analytické části spadá první kapitola, ve které jsou otestovány již existující nástroje na podporu konference. Druhá kapitola se pokouší odpovědět na otázku, zda je možné nástroj na podporu konference plnohodnotně nahradit redakčním systémem. Do analytické části spadá i kapitola třetí, ve které je vytvořen návrh aplikace pomocí UML diagramů, konkrétně diagramů případů užití.

Implementační část se zabývá tvorbou aplikace. Nejprve je ve čtvrté kapitole popsán výběr technologií, na kterých bude aplikace založena a je v ní uveden i jejich krátký popis. Poslední kapitola popisuje, jak byl nástroj vytvořen, implementaci jednotlivých vrstev MVC architektury a nakonec jsou uvedeny krátké ukázky funkčnosti aplikace.

Klíčová slova

Pořádání konference, UML, Java, framework, databáze

Abstract

This diploma thesis discusses the means that serve as a support for organizing conferences. It is divided in two parts; firstly an analytical part, secondly a part of the implementation. The analytical part belongs to first chapter, which is testing existence of the tools in support of the conference. The second chapter, is trying to answer the question whether it is possible tool in support of the conference to fully replace the content management system. In addition, the analytical part includes also the third chapter, in which the analysis is created by using UML diagrams, concerns cases of diagrams specifically.

The fourth section is mainly about the deals with the application . This chapter describes the selection of technologies on which the application are based,it also includes their explanations. The final chapter describes how the tool was created to implement each of its parts, and at last a short demonstration of its functions.

Keywords

Conference organization, UML, Java, framework, database

Poděkování

Děkuji Ing. Igoru Kopetschkemu, Ph.D. za vedení této diplomové práce. Dále bych rád poděkoval Ing. Milanu Křížovi a Ing. Zdeňku Kořínkovi i všem ostatním, kteří mě při psaní této práce podporovali.

Obsah

Seznam zkratek

CoMS	Conference Management System
OCS	Open Conference Systems
CMS	Content Management System
JPA	Java Persistence API
ORM	Objektově relační mapování
POJO	Plain Old Java Object
HQL	Hibernate Query Language
OGNL	Object-Graph Navigation Language

Seznam obrázků

1. Úvod

Odborná konference je, zjednodušeně řečeno, setkáním lidí, kteří se zajímají o dané téma. Ať je téma jakékoliv, všechny konference mají určité společné znaky - hlavní téma, možnost přihlášení zájemců, hlavní program, atd.

Organizace odborné konference je činnost, při které je potřeba synchronizovat množství lidí, kteří vystupují v různých rolích. Pro zjednodušení této synchronizace se v dnešní době využívají nástroje na podporu uspořádání odborných konferencí. Při jejich využití je organizace konference značně zjednodušena.

Kvalitní nástroj na podporu odborné konference by měl pokrýt celý životní cyklus konference - od jejího vytvoření, kde by měl obsahovat všechny potřebné informace pro potenciální zájemce, přes její řízení, ve kterém je potřeba již zmíněná synchronizace lidí, kteří vystupují v různých rolích, přijmutí jejich příspěvků, vygenerování a uveřejní programu, až po postkonferenční období, ve kterém se mohou objevit různé statistiky, fotogalerie, atd.

Proto vznikl požadavek na vytvoření takového nástroje i pro Technickou univerzitu v Liberci. Původním požadavkem bylo vytvoření nástroje na pořádání konferencí přesně pro TUL, který bude založený na platformě Java EE. Po jistých komplikacích bylo zadání upraveno na univerzální generátor konferencí.

Cílem této diplomové práce je tedy zanalyzovat již existující nástroje na tvorbu odborné konference a na základě této analýzy vytvořit nástroj vlastní.

2. Existující nástroje

Nástrojů na tvorbu a řízení konferencí není tak málo. Neexistuje standardizovaná zkratka pro tyto nástroje, proto pro účely této práce je používána zkratka CoMS - Conference Management Systems. Následující tabulka obsahuje většinu volně šiřitelných CoMS nástrojů, které jsou v současné době k dispozici.

Nástroj	**Práva náleží**	**Licence**	**Technologie**
Open Conference Systems	Public Knowledge Project	GNU GPL	php
Open Conference ware	Igal Koshevoy, Reid Beels	MIT	Ruby on Rails
A Conference Toolkit	Philippe Bruhat, Éric Cholet	GNU GPL	Perl
ConMan	Utah Open Source Conference	GNU GPL	Python
Pentabarf	DebConf	GNU GPL	Ruby 1.8.6
SCALEreg	SCALE	GNU GPL	Django framework, Python
Zookeepr	linux.conf.au	GNU GPL	Python (Pylons)
The MyReview System	Philippe Rigaux	GNU GPL	php, Zend Framework
HotCRP	Eddie Kohler	free	php
EasyChair	easychair.org	free	php
conf2py	Massimo Di Pierro	GNU GPL2	Python
ConfTool	University of Hamburg	open/shared source system	php
OpenConf	Zakon group	free	php

Tabulka 2.1: Tabulka již existujících nástrojů k pořádání konferencí

Všechny tyto nástroje mají společný úkol - nějakým způsobem fungovat jako podpora při pořádání odborné konference. Co se týče rozsahu podpory, nastávají značné rozdíly - existují aplikace, do kterých se pouze zadá text, který je pak staticky zobrazen na stránkách a tím podpora konference končí. Jedná se tedy o aplikace, které mají pouze informativní charakter. Existují však i velice sofistikované nástroje, které umožňují detailní generování konferencí, propracovanou správu uživatelů, jejich příspěvků, atd. Tyto nástroje by se daly rozdělit podle mnoha kritérií. Základní

rozdělení je však:

- zda je nástroj volně šiřitelný nebo komerční
- podle technologie, na níž byl vytvořený (php, java, ruby, perl, atd.)

Většina CoMS nástrojů nemá příliš velkou podporu ani dostatečnou dokumentaci, z níž by se daly vyčíst informace, které budou dále využity při analýze. Proto je potřeba alespoň některé nástroje nainstalovat a otestovat. CoMS jsou založeny na různých platformách, k jejich běhu je zapotřebí instalace mnoha balíčků (v případě systému Linux), frameworků či serverů. Na základě informací dohledatelných na stránkách nástrojů a diskuzních fórech je provedena instalace a test CoMS nástrojů, které by měly obsahovat všechny potřebné moduly pro podporu plnohodnotné konference.

2.1 Open Conference Systems

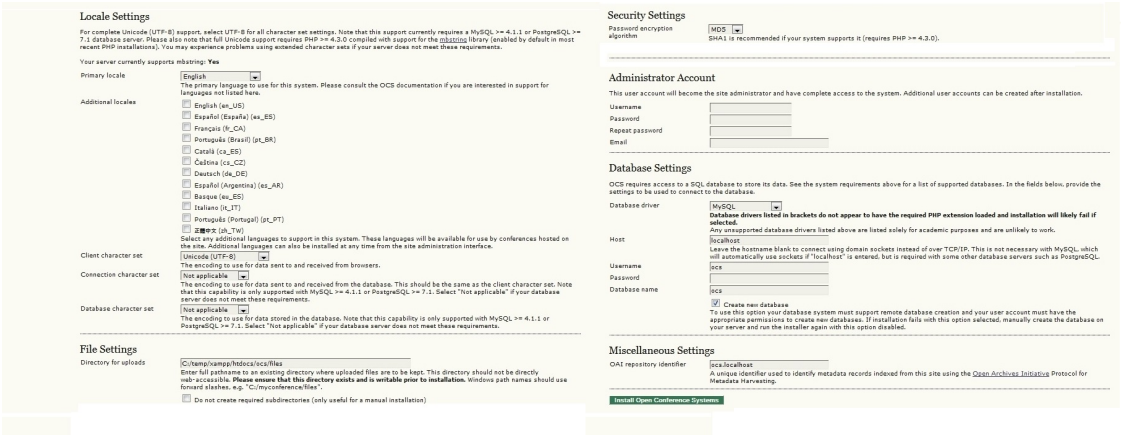
Open Conference System (OCS) je bezplatný webový publikační nástroj, který je schopen vytvořit webovou aplikaci sloužící jako kompletní podpora pro vědecké konference. OCS je open source řešením pro správu a publikování vědeckých konferencí. Jedná se o vysoce flexibilní publikační systém, který byl navržen tak, aby snížil čas a energii věnovanou administrativním a manažerským úkolům spojeným s organizací a řízením odborné konference. [9]

Práva na tento nástroj patří uskupení Public Knowledge Project, což je skupina převážně amerických univerzit. OCS spadá pod licenci GNU GPL[1]. Je založen na technologii php, proto je k jeho běhu zapotřebí server Apache. Jako databáze je použita MySQL. Nástroj OCS obsahuje všechny základní funkcionality, které je potřebné využít při tvorbě odborné konference. Jedná se například:

- vytvoření více konferencí
- registrace účastníků
- odeslání příspěvků na konferenci
- příjem abstraktů
- více verzí zaslaných příspěvků
- pokonferenční diskuze

Instalace OCS je uživatelsky přívětivá - stačí aplikaci spustit na serveru Apache. Při prvním spuštění nástroje se automaticky spustí jeho instalace, pomocí níž je nástroj nastaven do základního nastavení. Struktura instalačního menu je uvedena na obrázku 2.1.

[1] Licence, která vyžaduje, aby byla odvozená díla dostupná pod toutéž licencí.



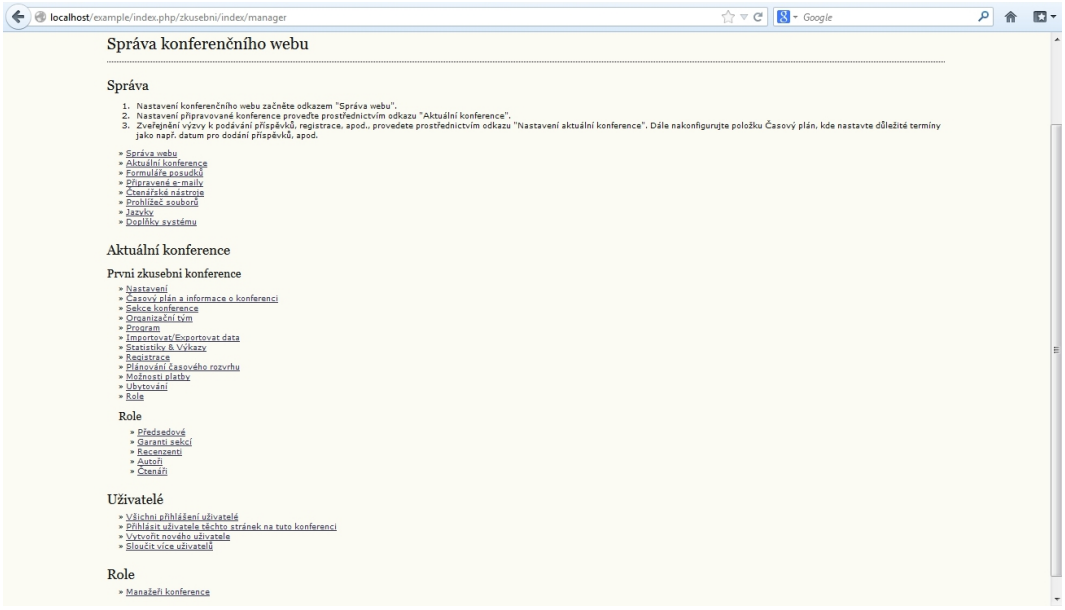
Obrázek 2.1: Instalace Open Conference Systems

Po instalaci je nutné, aby se uživatel přihlásil jako administrátor a donastavil aplikaci dle jeho požadavků. Administrátor stránek, tzv. superuživatel, se stará o globální nastavení aplikace, které je stejné pro všechny konference. Do tohoto nastavení patří mimo jiné text v hlavičce a v zápatí nástroje, jazykové nastavení, zabezpečení, atd. Superuživatel má neomezená práva, může libovolně zasahovat i do již vytvořených konferencí.

Po správném nastavení aplikace je možné začít vytvářet libovolné konference. Samotnou konferenci musí vytvořit superuživatel, ale následnou správu konference může přidělit uživateli, který je v roli tzv. manažera konference. Struktura rolí je v OCS poměrně propracovaná:

- administrátor aplikace - spravuje celou aplikaci, vytváří konference
- manažer konference - řídí konferenci, spravuje její účastníky, řídí příspěvky konference, stará se o archivaci konference
- garant konference - vytváří výzvu k podání příspěvků, přiděluje oponenty, na základně oponentury rozhoduje o přijetí či nepřijetí příspěvku
- autor - posílá příspěvek
- oponent - oponuje příspěvek
- účastník - účastní se konference jako divák
- čtenář - má právo pro přístup do některých sekcí aplikace

Při vytváření konference je prvním krokem její základní nastavení, které se provede v šesti jednoduchých krocích. V nich se nastavují licenční ujednání, viditelnost informací pro uživatele, kteří nejsou přihlášeni do systému, aktuální oznámení, záhlaví, zápatí, styly, přednastavené e-maily, apod. Po dokončení výše uvedeného nastavení se uživatel dostane do menu viz. obrázek 2.2.



Obrázek 2.2: Open Conference Systems - nastavení konference

Na obrázku 2.2 je vidět, že byla vytvořena konference s názvem *Prvni zkusebni konference*. V tomto menu se nyní nastaví zbylé informace o konferenci. Menu je rozděleno na dvě části - v sekci *Správa* se pouze upravuje nastavení z předchozích dvou kroků.

Důležitější je sekce *Aktuální konference*, ve které se nastavuje vše k vytvořené konferenci. Uživatel zde může mimo jiné nastavit místo konference, role uživatelů, spolupracující organizace, typy příspěvků (zda poslat abstrakty, plné práce nebo obojí), atd. Finální vzhled vytvořené konference je uveden obrázku 2.3.

Obrázek 2.3: Open Conference Systems - vytvořená konference

Na tuto konferenci je pochopitelně možné se zaregistrovat. Při registraci si může

uživatel zvolit role, ve kterých se chce konference zúčastnit. Může to být

- čtenář - pouze se zúčastní konference
- autor - zasílá příspěvek
- recenzent - zadá svůj obor a čeká na přidělení příspěvku

Uživatel může působit v několika rolích najednou. Pro testovací účely byla vybrána právě tato varianta. Po registraci a následném přihlášení má uživatel na výběr, zda vstoupí do

- menu pro autora - v něm má možnost zaslat příspěvek (buď pouze abstrakt nebo kompletní příspěvek, dle toho, jak je konference nastavena)
- menu pro recenzenta - může recenzovat příspěvek, pokud mu byl nějaký přidělen

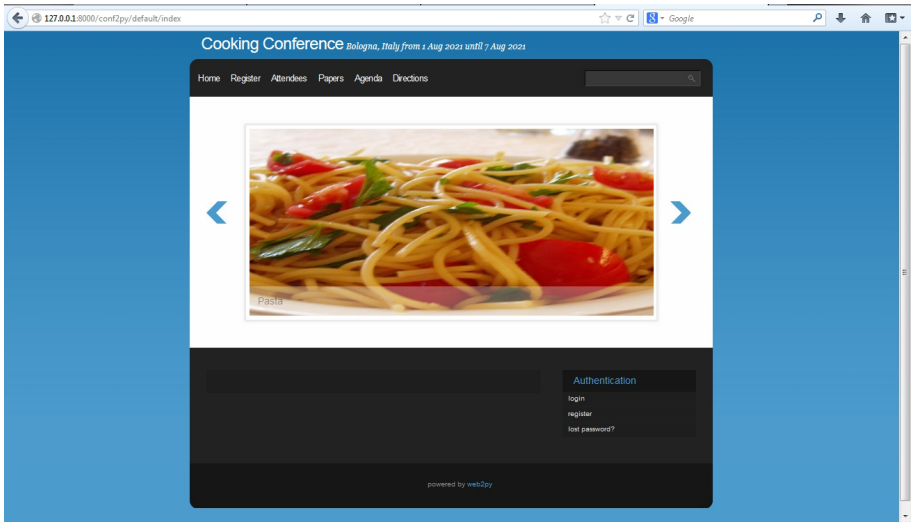
Příspěvky přiřazuje recenzentům garant konference. Do této role se nelze registrovat, ale vytváří ji buď administrátor aplikace nebo manažer konference. Po přiřazení konference proběhne recenzní řízení, ve kterém spolu recenzent práce a autor vzájemně komunikují. Výstupem recenzního řízení je to, že je buď příspěvek na konferenci přijat nebo nepřijat.

2.2 Conf2py

Conf2py je CoMS nástroj vytvořený v programovacím jazyce Python s pomocí frameworku Web2Py. Conf2py spadá pod licenci GNU GPL2[2]. Autorem je Massimo Di Pierro, resp. MetaCryption LLC. Tento nástroj je přepsanou verzí nástroje web2conf, což je nástroj, původně vyvinutý pro PyCon US 2009. Důvodem přepsání zdrojového kódu bylo zjednodušení a přidání funkcí. Podle dokumentace nástroj Conf2py obsahuje standardní moduly pro organizaci a řízení odborné konference jako jsou odesílání příspěvků, abstraktů, recenzní řízení, atd. Mimo to obsahuje i nadstandardní funkce jako jsou

- autentizace pomocí profilu Facebook, MySpace, atd.
- platba pomocí Authorize.Net
- tvorba slevových kuponů
- podpora LaTeX[2]

Aplikace je vytvořena pomocí frameworku web2py. Pro její běh je zapotřebí mít zprovozněný web2py server, na kterém je aplikace Conf2py spuštěna. Po zprovoznění serveru a spuštění aplikace se uživatel dostane na úvodní stránku, která je uvedena



Obrázek 2.4: Conf2py - úvodní menu

na obrázku 2.4. Úvodní stránka obsahuje všechny informace, které má CoMS nástroj splňovat.

Jsou to - domovská stránka se základními informacemi popisující konferenci, seznam účastníků s jejich profily, seznam prací, program využívající google kalendář a instrukce. Pro vytváření a řízení konferencí je potřeba se do aplikace registrovat v roli manažera konferencí.

Nástroj vypadá po grafické stránce velice atraktivně a dle jeho dokumentace i funkčně, avšak při jeho testování v rámci této diplomové práce nastala chyba, kterou nebylo možné odstranit. Při registraci do role manažera konference (aby bylo možné vytvořit testovací konferenci) se vyskytla chyba. Její výpis je uveden na obrázku 2.5.

Bohužel, přes veškerou snahu nebyla tato chyba odstraněna, takže nemohla být vytvořena zkušební konference, na které by byla ověřena funkčnost aplikace a modulů pro generování a podporu odborné konference. Testování tedy bylo ukončeno jako neúspěšné.

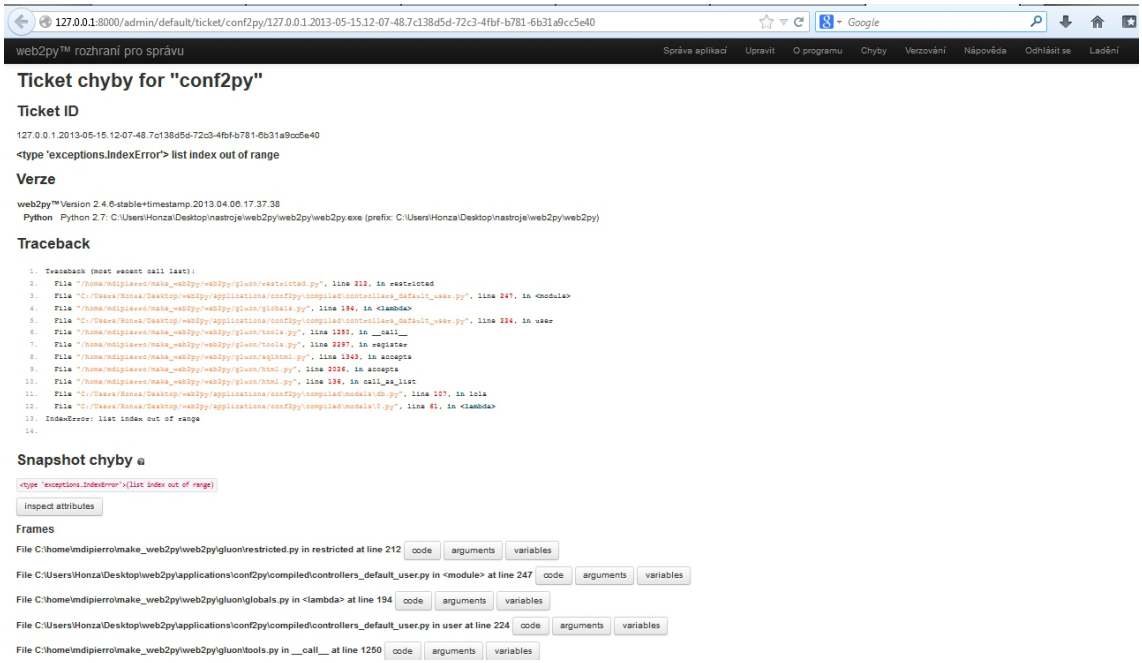
2.3 ConfTool

ConfTool je nástroj vytvořený týmem německých programátorů, práva na něj patří Univerzitě v Hamburku. Je vyvíjen ve dvou verzích. K nekomerčním a testovacím účelům je to verze VSIS ConfTool, kterou lze získat po splnění určitých podmínek[3]. Tato verze je limitována maximálním počtem 150 účastníků na vytvořené konferenci.

ConfTool je založen na technologii php, proto je pro jeho spuštění mít zapotření nainstalovaný server Apache. Pro ukládání dat je využita MySQL databáze. Instalace nástroje spočívá v nastavení konfiguračních souborů nástroje ConfTool, serveru

[2]Rozšíření GNU GPL

[3]V případě této diplomové práce to byl e-mail ze školního účtu, ve kterém byl vysvětlen důvod zájmu



Obrázek 2.5: Conf2py - chyba při registraci do aplikace

Apache a databáze MySQL podle návodu, který je získán společně s instalačním archivem.

Tento nástroj není klasickým CoMS nástrojem. CoMS nástroj by měl umožnit podporu odborné konference pomocí jedné komplexní aplikace, do které by měly být zadány všechny potřebné informace o konferenci. Nástroj ConfTool se vydal jinou cestou - poskytuje pořadateli odborné konference podporu pro řízení uživatelů a odesílání jejich příspěvků. To v praxi znamená, že nástroj neumožňuje tvorbu frontend[4] aplikace, který bude obsahovat všechny podstatné informace části aplikace, nýbrž pouze některé moduly backend[5]. Jsou to

- registrace uživatele podle úrovně
- odeslání příspěvku
- nastavení poplatků
- tisk faktury

Administrátorské rozhraní je uvedeno na obrázku 2.6. Podpora konference probíhá následovně:

1. je vytvořena informativní část konference, tato část je technologicky nezávislá od ConfTool
2. je nastaven nástroj ConfTool (viz výše)

[4]Slouží k označení části webu viditelné běžným návštěvníkům.
[5]Část aplikace sloužící k její administraci.

3. součástí nastavení je i vložení odkazu na konferenci
4. konference musí přesměrovávat registraci do ConfToolu

Obrázek 2.6: ConfTool - administrační menu

Příklad spolupráce mezi webovou aplikací a nástrojem ConfTool je uveden na obrázku 2.7.

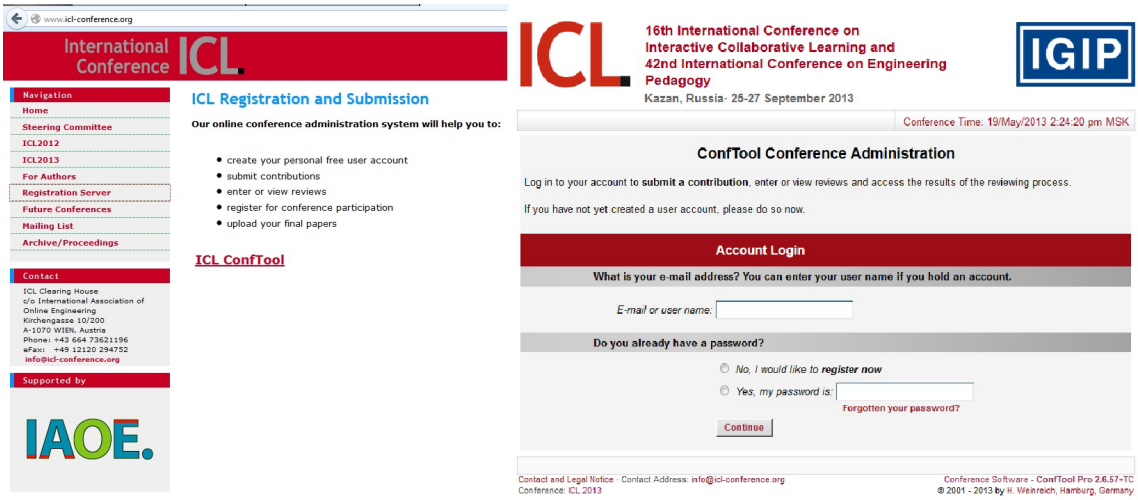
2.4 OpenConf

Posledním testovaným CoMS nástrojem je OpenConf. I tento nástroj je vyvíjen ve dvou verzích, a to v placené verzi OpenConf Professional a volně šiřitelné verzi OpenConf Community Edition. Volně šiřitelná verze je ochuzena o některé funkce. I tak je však plně dostačujícím nástrojem, který může sloužit jako podpora konference. Kromě obvyklých funkcí (řízení uživatelů, recenzní řízení, atd.) aplikace umožňuje:

- import a export do souborů formátu CSV / XML / XLS
- vícejazykovou podporu
- možnost využití CAPTCHA

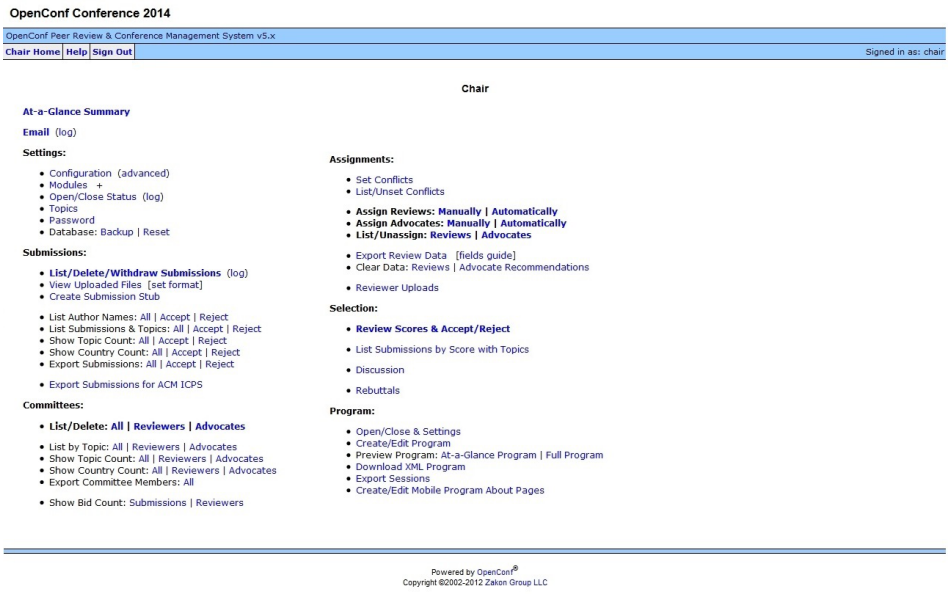
U OpenConf je opět je využita technologie php, tedy je třeba mít nainstalovaný server Apache. Data jsou opět ukládána do MySQL databáze. [10]

Soubory, které jsou potřebné k instalaci nástroje, je možné stáhnout na stránkách produktu. Na na těchto stránkách je i demo verze aplikace, která obsahuje testovací konferenci.



Obrázek 2.7: ConfTool - spolupráce ConfTool s webovou aplikací

Prvním krokem je vstup do administrátorské části aplikace, ve které je konference vytvořena. Tato sekce je uvedena na obrázku 2.8. Je v ní možné mimo jiné používat:



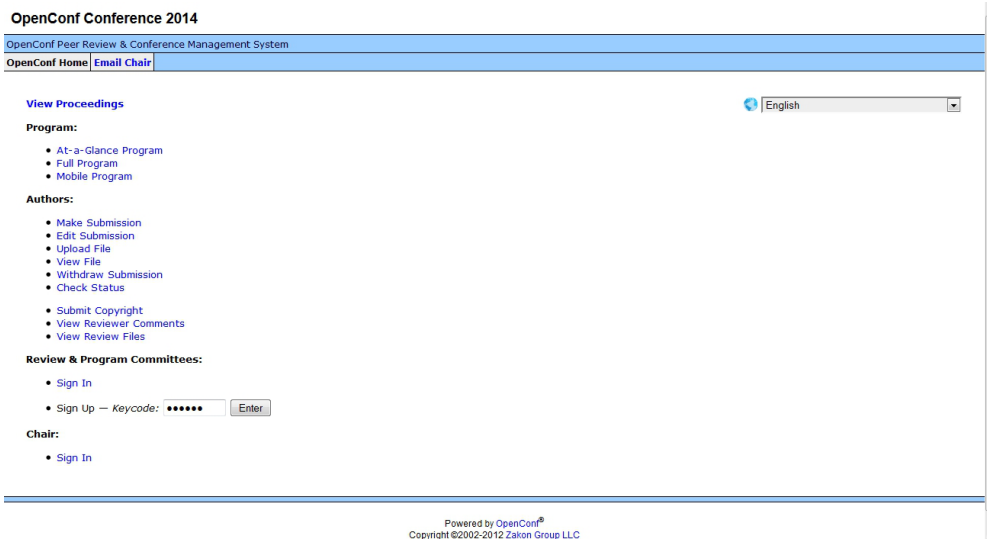
Obrázek 2.8: OpenConf - administrační menu

- kompletní a přehledné statistiky o uživatelích a přijatých i nepřijatých příspěvcích
- hromadné zasílání e-mailů vybraným skupinám uživatelů
- řízení přídavných modulů aplikace

- správu členů výborů
- tvorbu programu

Po vytvoření a nastavení konference se uživatel dostane na její úvodní stránka. Ta je zobrazena na obrázku 2.9, kde je vidět, že tato stránka poskytuje všechny důležité informace pro potenciální účastníky konference, mezi které patří:

- program
- možnost zaslání/editace příspěvku
- nahlížení do abstraktů a příspěvků
- registraci/login



Obrázek 2.9: OpenConf - úvodní stránka

2.5 Poznatky získané testováním nástrojů

Tabulka, která je uvedena na začátku této kapitoly obsahuje třináct tzv. CoMS nástrojů. Jak již bylo uvedeno, tyto nástroje jsou založeny na různých technologiích a platformách, mají nekvalitní podporu a dokumentaci, a proto je nejnáročnější částí při používání CoMS nástroje jeho instalace a nastavení. Původně měly být otestovány i nástroje Pentabarf, Hotcrp a Myreview, ale jejich instalace nebo nastavení neproběhly úspěšně.

Nicméně pomocí nástrojů, které byly úspěšně otestovány, byl získán přehled toho, co by CoMS aplikace měla splňovat. Mezi nezbytné součásti patří:

- backend část, pomocí které bude vytvořena a řízena konference

- frontend část s informacemi o pořádané konferenci
- možnost registrace uživatelů v různých rolích
- zasílání příspěvků/recenzní řízení
- rozdělení na část před/během/po konferenci

Bez výše uvedených funkcionalit by CoMS aplikace postrádala smysl. Jako rozšiřující funkce by mohla obsahovat:

- jazykové mutace
- podporu platebních bran (např. PayPal)
- větší možnosti grafického nastavení
- fotogalerii

3. CMS systémy

V předchozí kapitole byly vybrány a otestovány některé CoMS nástroje. Tato kapitola se pokusí objasnit, zda by bylo možné využít místo CoMS nástroje nedal jako podpora konference využít nějaký z redakčních systémů.

Redakčním systémem (CMS) se nazývá zabezpečené rozhraní určené primárně k editaci obsahu webových stránek. Webovou prezentaci lze v podstatě rozdělit na dvě linie, grafickou a obsahovou. Grafickou linii obstarává předem definovaná šablona, která definuje vzhled prezentace. Obsah stránek se načítá většinou z databází, a ten můžeme pomocí k tomu učenému nástroji, téměř libovolně měnit.

Moderní redakční systémy nepředstavují pro běžného provozovatele žádnou zátěž v podobě znalostí programovacích jazyků. V podstatě jedinou podmínkou pro editaci webu prostřednictvím redakčního systému je základní dovednost při práci s počítačem. [20]

Mezi základní funkce CMS patří

- Tvorba, modifikace a publikace dokumentů (článků) zpravidla prostřednictvím webového rozhraní, často s využitím jednoduchého online WYSIWYG editoru[1] nebo jednoduchého systému formátování textu
- řízení přístupu k dokumentům, zpravidla se správou uživatelů a přístupových práv
- správa diskusí či komentářů, ať už k publikovaným dokumentům nebo obecných
- správa souborů
- správa obrázků či galerií
- kalendářní funkce
- statistika přístupů

[14]

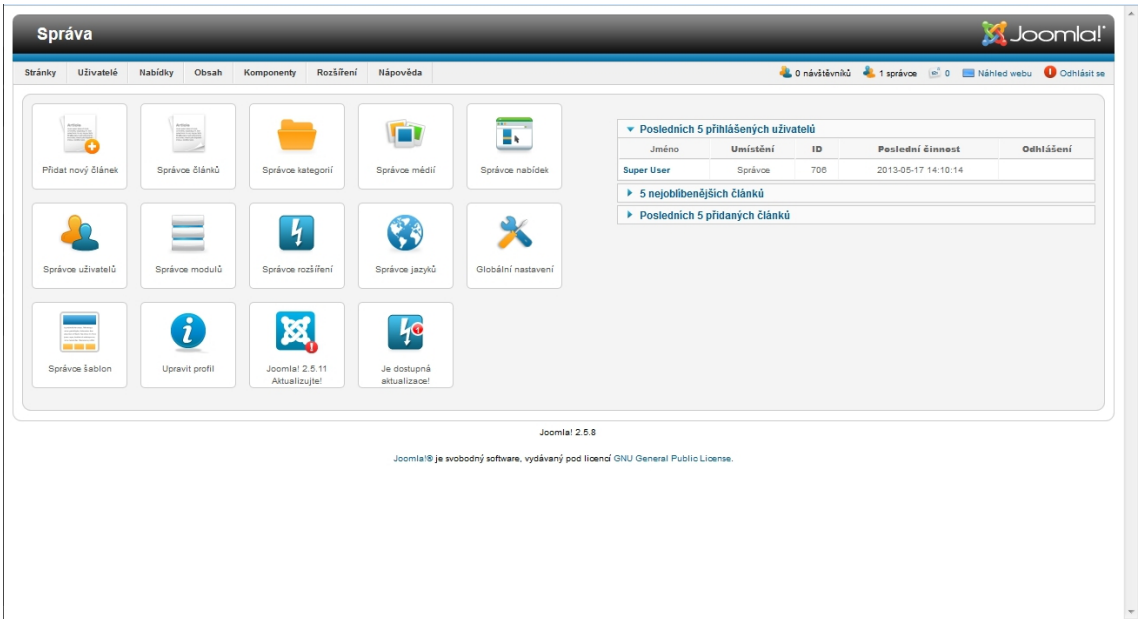
[1] Editace, při kterém je verze zobrazená na obrazovce vzhledově totožná s výslednou verzí dokumentu.

3.1 CMS systém jako podpora odborné konference

V předchozí sekci byly stručně popsány CMS systémy a jejich výhody. Poslední dobou tyto nástroje nabývají na oblibě. Proto se nabízí otázka, zda by bylo možné použít univerzální CMS nástroj. Pro zjištění odpovědi na tuto otázku je vytvořena webová aplikace na podporu odborné konference pomocí CMS nástroje Joomla.

Joomla je bezplatný open source CMS pro účely publikování informací na internetu a intranetu. Je napsána v jazyce PHP a využívá databázi MySQL. Joomla podporuje caching, indexaci stránek, RSS, tisknutelné verze stránek, zobrazování novinek, blogy, diskusní fóra, hlasování, kalendář, vyhledávání v rámci webserveru, lokalizace a vícejazyčné verze. [6]

Instalace Joomly na uživatelovu doménu je díky kvalitní dokumentaci jednoduchá. Po instalaci a nastavení nástroje se uživatel dostane do administrátorské části, která je uvedená na obrázku 3.1. Oproti CoMS nástroji již zde nastává pro běžného uživatele komplikace, která by ho v nástroji na tvorbu konferenčního webu nepotkala - v konferenčním nástroji jsou předpřipravé položky pro vytváření odborné konference. Kdežto v CMS systému, kvůli jeho univerzálnosti, musí potenciální manažer konference celé menu vytvořit sám. Už tento krok by pro někoho mohl být překážkou.



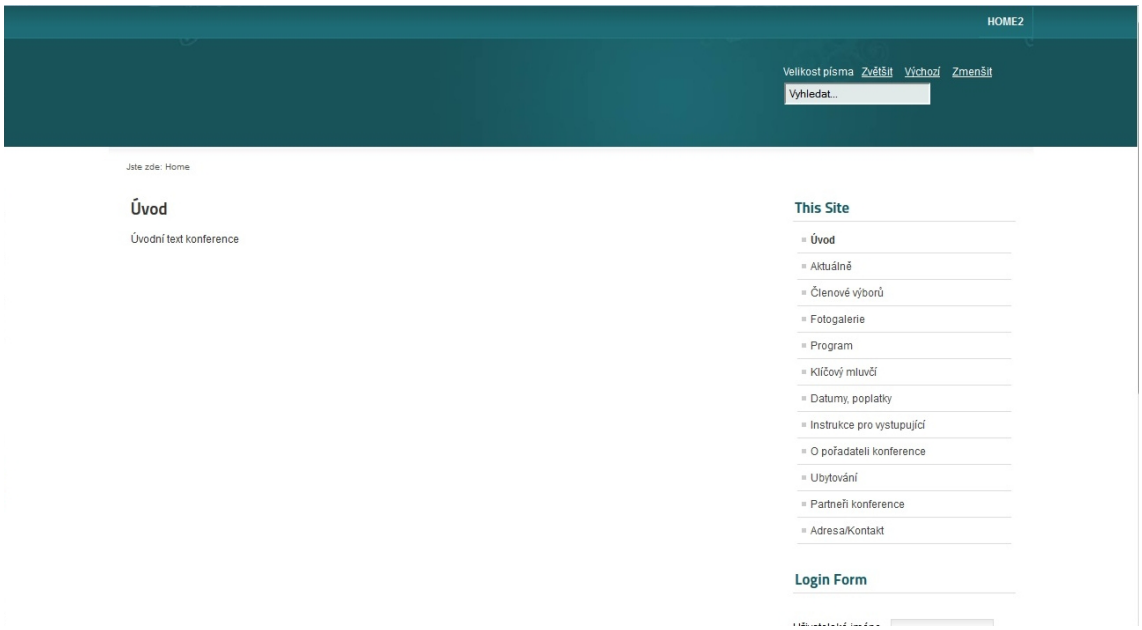
Obrázek 3.1: Administrační část Joomly

Nicméně pokud je uživatel schopen vybrat šablonu a vzhled stránky, získává oproti CoMS značnou výhodu. Drtivá většina CoMS nástrojů má pouze malé možnosti nastavení vzhledu stránky a většinou si jsou vytvořené stránky konferencí podobné. Kdežto při použití CMS má možnosti výběru vzhledu mnohonásobně vyšší.

Druhým krokem po výběru šablony je tvorba menu. Zde nastává pro CMS nástroj

další komplikace, a sice, že uživatel musí sám vytvořit menu konference. Při použití CoMS nástroje by tento problém odpadl, protože v CoMS nástrojích jsou položky menu pevně dané takže by do nich uživatel pouze zadal potřebná data.

Pokud se dostane i přes tento problém, může se mu povést vytvořit frontend část aplikace. Na základně informací z předchozí kapitoly bylo vytvořeno menu a následně byly vloženy informace pro každou položku z menu. Výsledný vzhled je uveden na obrázku 3.2. Vytvořená konference je vizuálně příjemná a vložené informace lze kdykoliv editovat.



Obrázek 3.2: Konference vytvořená pomocí CMS Joomla

Pomocí výše uvedeného postupu byla pomocí nástroje Joomla vytvořena frontend část aplikace. K tomuto účelu Joomla poslouží více než dobře. Problém nastává, pokud by chtěl tvůrce konference využít některou ze specializovaných funkcionalit, které jsou součástí CoMS nástrojů (např. recenzní řízení, konferenční role, atd.).

Pokud je na problematiku nahlíženo z pohledu uživatele, který má jen základní znalosti práce s počítačem, resp. nemá znalosti z odvětví programování, je pro něj téměř nemožné, aby pomocí CMS nástroje byl schopen vytvořit aplikaci, která bude fungovat jako plnohodnotná podpora odborné konference.

Jako možná varianta použití CMS nástroje jako plnohodnotné podpory při pořádání odborné konference je využití nástroje ConfTool. Pokud by uživatel tento nástroj využil v kombinaci s jeho stránkou vytvořenou v CMS, tak by plnohodnotná podpora měla fungovat.

Závěrem této kapitoly lze říci, že pokud se pořadatel konference spokojí s "informativní" funkcí stránek podporujících jeho konferenci, pak je možné využít libovolný redakční systém. Pokud ovšem uživatel vyžaduje, aby jeho aplikace splňovala alespoň nějaké funkce z backend části aplikací na podporu konferenčních webů, pak bez znalosti programování, pomocí něhož by si dopsal potřebné funkce, není schopen

pouze s využitím CMS nástroje této funkčnosti dosáhnout.

4. Návrh aplikace

Na konci první kapitoly byly shrnuty nezbytné funkcionality, které by měl plnohodnotný CoMS nástroj splňovat. V této kapitole je na základě tohoto souhrnu vytvořena analýza a návrh aplikace. K tomu bude využit jazyk UML[1], konkrétně diagramy případů užití pro zaznamenání uživatelských požadavků a ERD diagramy pro vytvoření návrhu databáze.

4.1 Webová aplikace na podporu odborné konference

Jedním z hlavních cílů CoMS aplikace je informovat potenciálního zájemce o konferenci o jejích detailech. Proto by měla aplikace určitě obsahovat:

- úvodní stránku obsahující základní informace o konferenci
- tematické okruhy konference
- stránku se členy organizačního a vědeckého výboru
- program
- seznam klíčových mluvčích
- informace o důležitých termínech a poplatcích
- instrukce pro autory
- partnery konference
- kontakt
- registraci uživatele

Výše uvedený seznam zahrnuje požadavky z pohledu uživatele, který se chce na konferenci přihlásit - frontend část aplikace. Druhou, velmi důležitou součástí aplikace je však i zadní část aplikace, tzv. backend. Do něj by mělo patřit:

- vytvoření a nastavení konference
- registrace a přihlášení uživatelů, více uživatelských rolí

[1] Grafický jazyk pro vizualizaci, specifikaci, navrhování a dokumentaci programových systémů.

- příjem a řízení uživatelských prací
- odeslání abstraktu
- recenzní řízení

4.2 Zpracování požadavků

Prvním krokem analýzy je obvykle zpracování požadavků. Cílem zpracování požadavků je nalezení, co by daná aplikace měla splňovat. Nejedná o žádnou hlubokou analýzu, ale spíš o určení toho, co má daná aplikace mít za funkce. Jinak řečeno - neřeší se, *jak* má daná aplikace něco udělat, ale pouze to, *co* má udělat. Existují dva základní typy požadavků:

- funkční - specifikují požadavky na funkčnost systému
- nefunkční - specifikují vlastnosti systému, případně podmínky omezující fungování systému

Zdroje funkčních požadavků lze identifikovat zejména v požadavcích zákazníka, legislativě, již existujících systémech, vlastního know-how, hardwarové a softwarové vybavenosti, atd.

Nefunkční požadavky obsahují vlastnosti systému a jeho případná omezení. Dále určují způsob, jímž bude systém implementován. Mezi nefunkční požadavky patří např. dodržení určitých standardů, využití určených komponent, rychlost odezev systému na specifické operace, nároky na výkonnost systému, zabezpečení systému, použitá architektura, atd. [19]

4.2.1 Identifikace funkčních požadavků

Tato aplikace bude obsahovat několik systémů[2]. Prvním systémem je tzv. *administrátorský systém*. Do něj má vstup pouze administrátor celé aplikace. Tento uživatel je pro celou aplikaci pouze jeden. V *administrátorském systému* je možné spravovat všechny vytvořené konferece.

Druhým systémem je *manažerský systém*. Do tohoto systému může vstoupit každý uživatel, který se registruje jako manažer konference. V tomto systému je možné vytvářet a spravovat konference. Každý manažer vytváří a spravuje pouze svůj seznam konferencí.

Posledním systémem je *uživatelský systém*. Ten se dělí na již zmiňovanou frontend část aplikace, ve které jsou obsaženy informace o vytvořené konferenci a backend část, do které může vstoupit pouze registrovaný uživatel, ve které je možné editovat svůj profil a řídit odesílání příspěvků na konferenci.

[2] Systémem se v UML nazývá vymezená část aplikace do které má aktér přístup.

4.2.2 Identifikace nefunkčních pozadavků

Nefunkční požadavky by se daly shrnout v následujícím seznamu:

- aplikace bude založena na platformě Java EE
- aplikace bude využívat frameworky *Struts*[2] a Hibernate
- data budou uložena v databázi MySQL

4.3 Případy užití

Využití diagramů případů užití[3] je jednou z možností, jak graficky zaznamenávat požadavky. Tento způsob byl vybrán pro jeho jednoduchost a současně dobrou názornost. Aplikace se skládá z různých systémů, do kterých vstupuje několika uživatelských rolí, přičemž různé role mají různá práva pro vstup do jednotlivých systémů.

4.3.1 Uživatelské role

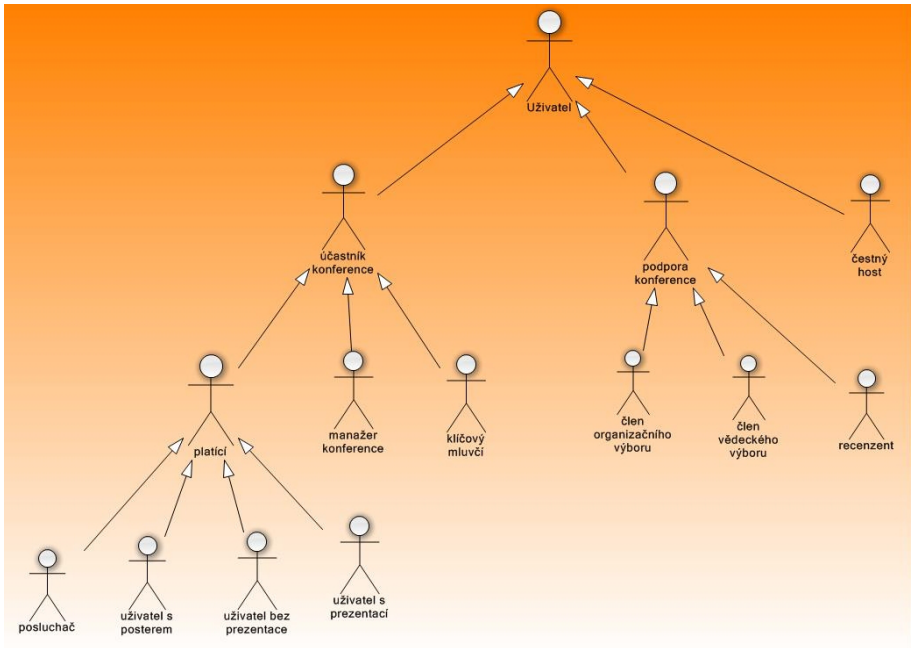
- administrátor aplikace - je pouze jeden, spravuje všechny vytvořené konference
- manažer aplikace - vytváří a spravuje svoje konference
- člen organizačního výboru, člen vědeckého výboru - patří do týmu, který zajišťuje konferenci
- role, které se nepodílejí na organizaci konference
  - posluchač
  - uživatel s prezentací posteru
  - uživatel, který se chce zúčastnit prezentací posterů
  - uživatel, který má příspěvek, ale nechce ho prezentovat
  - uživatel, který má příspěvek a chce ho prezentovat
- klíčový mluvčí - kapacita ve svém oboru, jeho příspěvek je automaticky přijat
- čestný host - pozvaní hosté
- recenzent - provádí recenzování příspěvků

Všichni aktéři jsou odvozeni od role uživatel. Ta vzniká vždy při vložení uživatele do aplikace a jsou pomocí ní uloženy základní informace o uživateli. Důležitou vlastností by mělo být, aby se jednotlivé role mohly prolínat, resp. aby uživatelé mohli vystupovat ve více rolích bez nutnosti vytváření více účtů.

[3]Diagram zachycuje vnější pohled na modelovaný systém a tím pomáhá odhalit hranice systému a slouží jako podklad pro odhady rozsahu.

Současně je ale dobré zamezit ”střetu zájmů” tak, aby např. uživatel, který poslal svůj příspěvek do recenzního řízení, mohl vystupovat v roli recenzenta a tím sabotovat příspěvky ostatních uživatelů, kteří se taktéž snaží projít recenzním řízením, atd.

Proto byly role rozděleny na dvě základní skupiny. Na role sloužící jako *podpora konference*, které mají volný vstup, mohou působit jako recenzenti a současně např. jako klíčový mluvčí či členové výborů a role, které se nepodílejí na organizaci konference, to jsou tzv. *účastníci konference*. To jsou role, do kterých patří všichni, kteří se účastní konference buď s příspěvkem, posterem nebo jako posluchač. Struktura rolí je uvedena na obrázku 4.1.



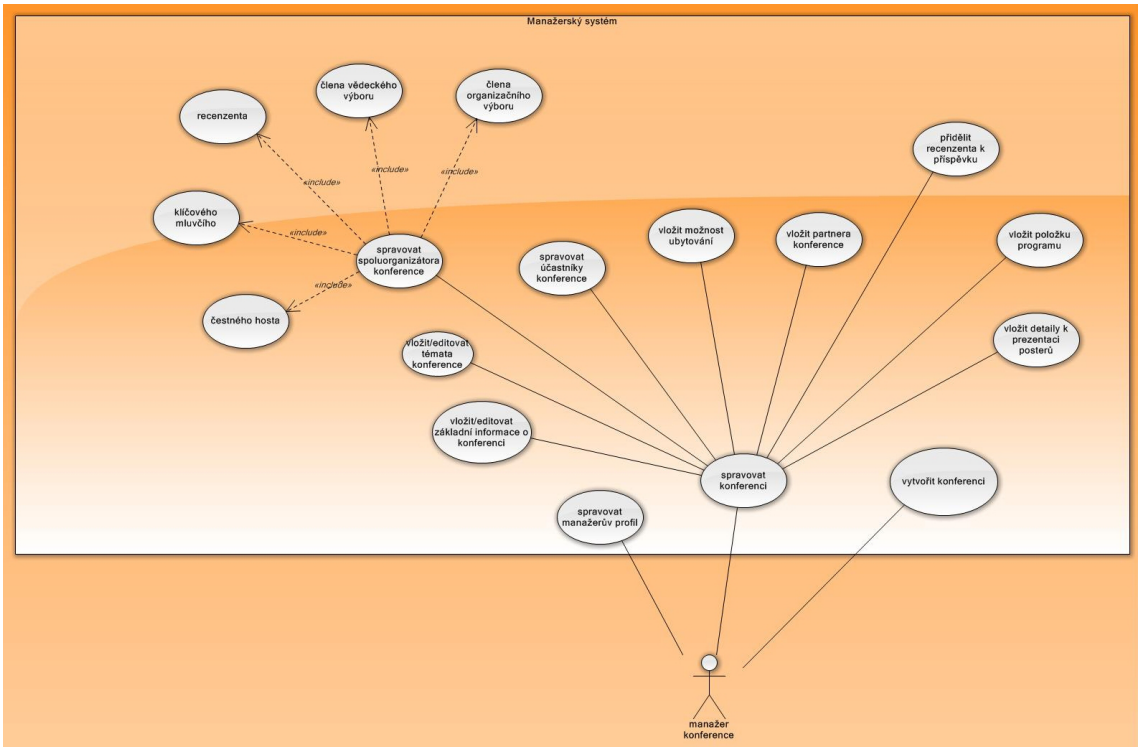
Obrázek 4.1: Struktura uživatelských rolí

4.4 Diagramy případů užití

Jak již bylo uvedeno, aplikace je rozdělena do tří základních uživatelských systémů. Nejjednodušším z nich je *administrátorský systém*. Pro tento systém není potřeba pomocí diagramů případů užití modelovat, protože se jedná pouze o jednoho uživatele, který vstoupí do systému.

Složitějším je *manažerský systém*, který je uvedený na obrázku 4.2. Do tohoto systému má právo vstoupit pouze manažer konference. Ten má právo konferenci vytvořit a následně ji spravovat.

Třetím systémem je *uživatelský systém* (viz obrázek 4.3). Do tohoto systému může vstoupit každý uživatel, který je přihlášený na danou konferenci. Uživatel se buď registruje jako *účastník konference* nebo jako *podpora konference*. Pokud se



Obrázek 4.2: Manažerský systém

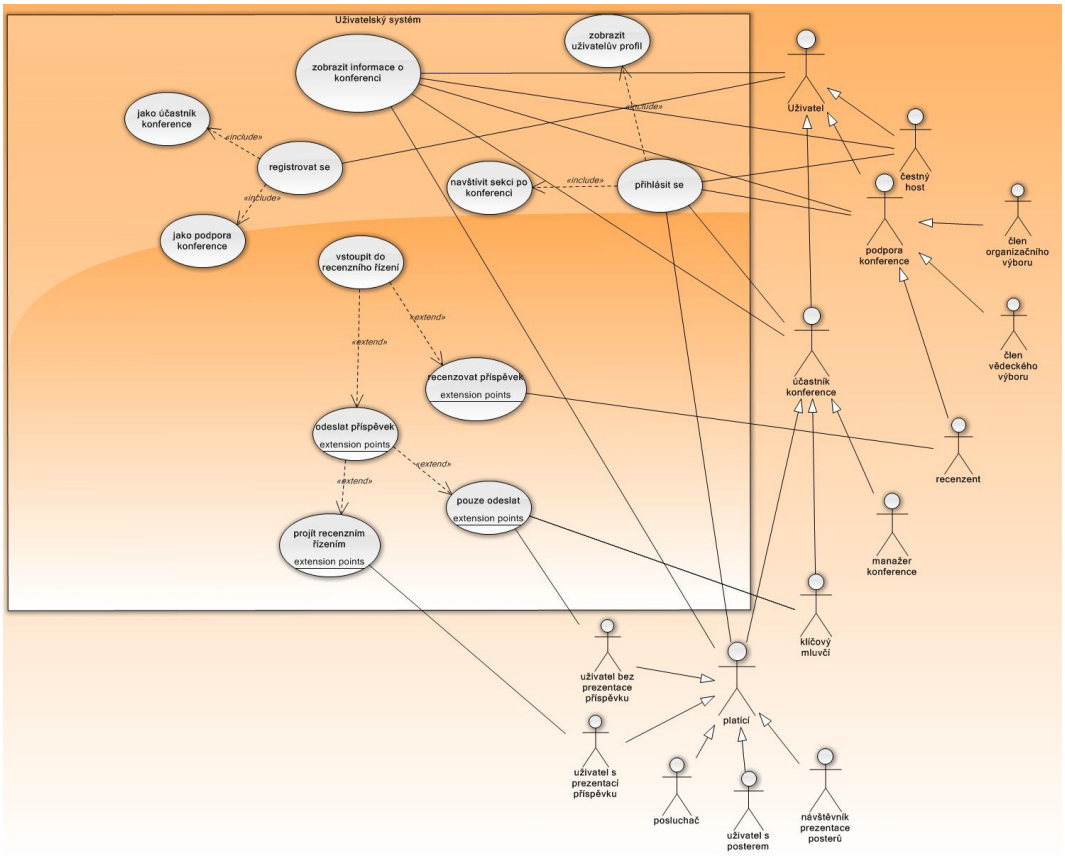
registruje jako *účastník konference*, je buď členem *organizačního* nebo *vědeckého* výboru nebo *recenzentem*. Tato registrace je zdarma.

Pokud se registruje jako *platící účastník* (*návštěvník prezentace posterů*, *posluchač*, *uživatel s posterem*, *uživatel s příspěvkem bez jeho prezentace*, *uživatel s prezentací příspěvku*), je nucen zaplatit vstupní poplatek do uvedeného data. Role *uživatel s příspěvkem bez prezentace* a *uživatel s prezentací příspěvku* zasílají do aplikace svoje příspěvky. První z uvedených si nepřeje svůj příspěvek prezentovat, proto ho pouze zašle a příspěvek neprochází recenzním řízením. Druhý typ role posílá svůj příspěvek do recenzního řízení, kterým pokud projde, bude práce prezentována na dané konferenci.

4.5 Návrh databáze

Databáze je nepostradatelnou součástí této aplikace. Kromě pár statických textů obsažených přímo v JSP stránkách, budou v databázi uložena všechna data, která bude aplikace využívat.

Návrh databáze, stejně jako diagramy případů užití, vyplývá ze zpracovaných požadavků na aplikaci. Obecně byla databáze navržena tak, že pro každou Java třídu byla vytvořena jedna tabulka a pro každou proměnnou jeden sloupec v databázi. Špatně navržená databáze může v průběhu implementace aplikace způsobit značné problémy, avšak v případě využití architektury MVC je případné rozšíření či úprava již navržené databáze poměrně jednoduchá. Schéma databáze je uvedené na



Obrázek 4.3: Uživatelský systém

obrázku 4.4. Toto schéma vymodelováno pomocí ERD diagramu[4]. Všechny návrhy byly vytvořeny v nástroji Software Ideas Modeler.

V rámci co nejvyšší dosažené normální formy muselo být mezi tabulkami použito několik typů vazeb. Při správném použití je databáze tzv. normalizovaná, což znamená, že je pružná, neobsahuje duplicitní data a je jednoduše udržovatelná. Těmto vazbám se říká kardinalita vztahu. Dělí se na tři skupiny:

- 1:1 – vztah, ve kterém na obou stranách vystupuje pouze jeden objekt dané entity
- 1:n – na jedné straně je pouze jeden objekt, který je ve vztahu s jedním nebo více objekty na straně druhé
- m:n – vztahy, ve kterých vystupuje více objektů na obou stranách

[4]Nástroj pro datové modelování, podmnožina jazyka UML.

Obrázek 4.4: Návrh databáze pomocí ERD

5. Technologie

Před implementací návrhu z předchozí kapitoly je nejprve nutné vybrat technologii, pomocí které bude implementace provedena. V dnešní době existuje poměrně široké spektrum technologií a nástrojů pro tvorbu webových aplikací. Mezi nejpoužívanější patří:

- php
- perl
- asp
- jsp
- python

Aplikace na tvorbu a podporu odborné konference spadá spíš do kategorie informačních systémů, než že by se jednalo o "obyčejné" webové stránky, které mají pouze informativní charakter. Informační systém (IS) je soubor lidí, technologických prostředků a metod, které zabezpečují sběr, přenos, zpracování a uchování dat za účelem tvorby prezentace informací pro potřeby uživatelů. Příkladem informačního systému může být kartotéka, telefonní seznam, kniha došlé pošty nebo účetnictví. [5]

Z výše uvedeného seznamu technologií by se pro tvorbu CoMS aplikace dala pravděpodobně využít většina z nich. Během studia na Fakultě mechatroniky však autor prošel několika kurzy programování v jazyce Java, a proto byla využita technologie založená právě na tomto objektově orientovaném programovacím jazyce. Bude tedy použita technologie JavaServer Pages a k ní použitelné webové frameworky.

5.1 Java EE

Java EE je součástí platformy Java, která se používá pro tvorbu informačních systémů. Jedná se o vývoj, který spojuje výhody internetových technologií a technologií používaných v rozsáhlých podnikových systémech. Mezi další výhody patří:

- nezávislost na operačním systému klientů i serverů
- databázová nezávislost

- nezávislost na aplikačních serverech
- automatická podpora bezpečnostních služeb
- možnost vzdálené správy aplikací přes internet
- možnost dynamického vyvažování zátěže serverů
- snadná výměna částí aplikace bez přerušení provozu

[11]

5.2 Výběr frameworku

Frameworkem se rozumí sada funkcí, tříd a nástrojů, které někdo (zpravidla komunita) vytvořil pro ostatní programátory. Obsahují sady tříd pro práci s databází, komunikaci pomocí HTTP protokolu, paginaci a spoustu dalších věcí. Moderní knihovny navíc obsahují "kostru" webové aplikace založené na architektuře MVC (Model-View-Controller), která odděluje data, výstupy pro uživatele a vlastní řídící program. Hlavními výhodami při využití frameworku je úspora času (nákladů) a široká podpora (u správně zvoleného frameworku). [18]

Java web frameworků existuje nepřeberné množství. Pro informační systém je vhodné použít framework založený na architektuře MVC. Základním předpokladem pro tuto aplikaci je, aby byl framework open source[1]. Open source frameworků využívajích javu je široká a neustále se měnící škála. Open source framework, jak už vyplývá z jeho názvu, může vyvíjet každý a od toho se odvýjí i jeho kvalita a rozsah - od poměrně malých a téměř nepoužívaných projektů až po rozsáhlé projekty s masivní členskou podporou a kvalitní dokumentací.

Hodnocení frameworků je komplikovanou činností a není cílem této práce. Po prozkoumání několika zdrojů zabývajících se touto tématikou a konzultací s vedoucím práce byl vybrán framework *Struts*[2]. Důvod jeho výběru, resp. jeho výhody jsou uvedeny v kapitole 5.5.

5.3 Nástroje pro běh aplikace

Ještě před implementací aplikace je potřeba nainstalovat a nakonfigurovat nástroje, které jsou k běhu aplikace nezbytné. Jsou to:

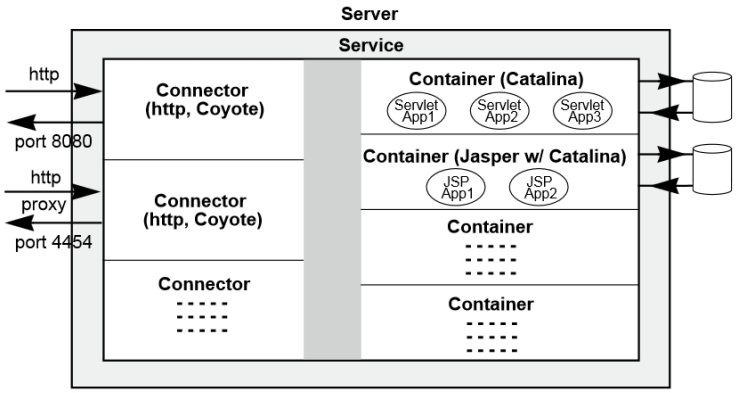
- Apache Tomcat
- Maven
- MySQL server

[1] Software s otevřeným zdrojovým kódem.

5.3.1 Apache Tomcat

Apache Tomcat je open-source webový server a servletový kontejner. Implementuje specifikace pro JSP (Java server pages) a Java servlety a poskytuje webové prostředí pro spouštění jazyku Java. [1] Konfigurace Tomcatu je prováděna pomocí XML souboru. Tomcat se skládá se tří modulů:

- Catalina - je servlet kontejner. Implementuje specifikace pro servlety a Java-Server Pages.
- Coyote - je HTTP connector, který podporuje protokol HTTP 1.1 pro webový server nebo aplikační kontejner. Coyote "poslouchá" TPC port, pošle požadavek do Tomcatu a následně odpoví na požadavek klientovi.
- Jasper - tzv. engine Tomcatu, kompiluje JSP soubory na servlety [12]



Obrázek 5.1: Struktura Tomcatu [12]

5.3.2 Maven

Apache Maven je nástroj pro správu, řízení a automatizaci buildů aplikací. Základním principem fungování Mavenu je popsání projektu pomocí Project Object Model (fyzicky se jedná o XML soubor), který popisuje softwarový projekt nejen z pohledu jeho zdrojového kódu, ale i včetně závislostí na externích knihovnách, popisu procesu buildování a různých funkcí s tím spojených (jako je například spouštění testů). Maven je postaven na modulární architektuře a funguje na principu volání jednotlivých pluginů, přičemž samotný Maven pouze obstarává dodání a spuštění nadefinovaných pluginů. Maven nemá žádné vlastní grafické uživatelské rozhraní a běží pouze v přikazovací řádce. Výhody použití Mavenu:

- Nemusejí se stahovat knihovny z internetu ručně, stačí přidat do *pom.xml* závislost, která se automaticky stáhne a propojí s projektem
- Maven definuje strukturu projektu, pomocí které se v něm programátor lépe orientuje [4]

5.3.3 MySQL databáze

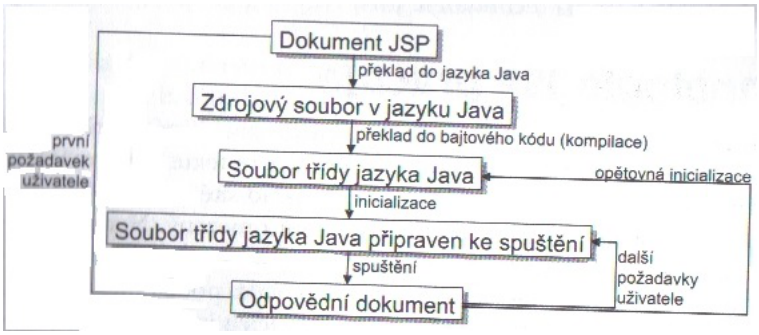
MySQL je multiplatformní relační databáze. Pro svou snadnou implementovatelnost (lze jej instalovat na Linux, MS Windows, ale i další operační systémy), výkon a především díky tomu, že se jedná o volně šiřitelný software, má vysoký podíl na v současné době používaných databázích. [7]

MySQL databáze je součástí balíčku XAMPP, jehož instalace značně zjednodušuje obsluhu databáze. Součástí balíčku XAMPP je i vývojové nástroj phpMyAdmin (k jehož spuštění je zapotřebí server Apache), pomocí něhož se jednoduše k nainstalované databázi přistupuje.

MySQL je tzv. relační databáze, což znamená, že se data ukládají do tabulek a musejí být atomická. Objekt, na kterém je založeno objektově orientované programování, však může obsahovat více informací (není atomický), a proto se musí vyřešit problém, jak uložit objekt do dabáze. To se řeší pomocí tzv. perzistence dat. Více viz sekce 5.6.

5.4 JSP - JavaServer Pages

JSP je jedna z několika technologií, které lze použít při tvorbě dynamických webových stránek. JSP je veskrze serverová technologie. Z toho vyplývá, že data dokumentu JSP, přístup k databázi, stejně jako další součásti vyžadující zásah ze strany serveru, může sdílet více uživatelů.



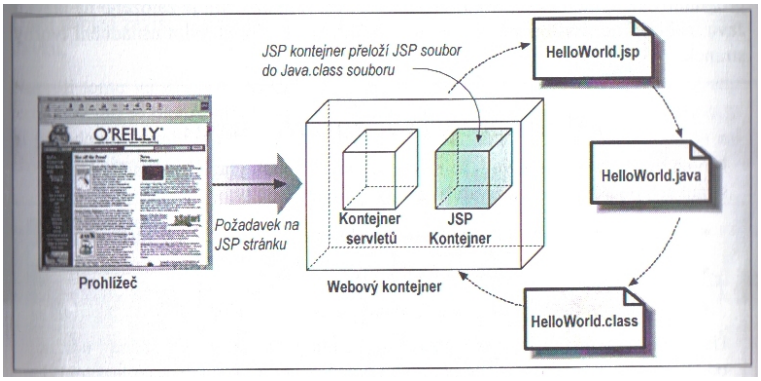
Obrázek 5.2: Zpracování JSP [17]

Na rozdíl od jiných technologií však JSP umožňuje plně využít síly programovacího jazyka Java. Výhodou JSP vůči ostatním technologiím je to, že JSP není skriptovací jazyk, takže dokument není při každém požadavku interpretován úplně od začátku. Při prvním použití je JSP dokument přeložen do bajtového kódu jazyka Java - do binárního formátu s možností rychlého a efektivního spouštění kódu. Způsob zpracování JSP kódu nejlépe vysvětlí obrázek 5.2.

JSP je přirozeným rozšířením technologie Java Servlet. Pomocí překladače se z JSP stávají Java servlety. JSP jsou textové dokumenty s příponou .jsp obsahující kombinaci statických HTML tagů, tagů ve stylu XML a scriptletů. Tagy a scriptlety zapouzdřují logiku tvorby obsahu stránek. Soubory .jsp se předem zpracují a

převedou na soubory s příponou .java. V tom okamžiku kompilátor Javy zkompiluje zdrojový kód a vytvoří .class soubor, který může být spuštěn kontejnerem servletů.

Překladač, který převede .jsp soubor na .java soubor se stará o vše, co souvisí s vytvořením Java servletu z JSP. Průběh kompilace je vysvětlen na obrázku 5.3



Obrázek 5.3: Kompilace JSP na Java servlet [17]

Technologie JSP se stala velmi oblíbeným řešením pro tvorbu webových aplikací založených na platformě Java. JSP nabízí oproti své konkurenci několik podstatných výhod:

- JSP je specifikace, nikoliv produkt. Vývojáři si mohou vybrat nejvhodnější přístup.
- JSP stránky se kompilují a ne interpretují, čímž se zlepšuje výkon
- JSP stránky podporují jak skriptování, tak přístup k plnohodnotné Javě a lze je rozšířit pomocí uživatelských tagů
- JSP stránky sdílejí charakteristiku *Write Once, Run Anywhere* z technologie Java.

JSP stránky představují společně se servlety výhodnou alternativu k jiným typům programování dynamických webových stránek. Jelikož obojí je založeno na jazyku Java, nabízejí nezávislost na platformě, rozšiřitelnost a především usnadnění tvorby stránek. [17]

5.5 *Struts*^{2}

Samotné JSP na tvorbu aplikace nestačí, proto byl použit framework *Struts*^{2}. Framework *Struts*^{2} je navržen tak, aby optimalizoval celý vývojový cyklus, od buildingu, přes deployment na server až po údržbu.

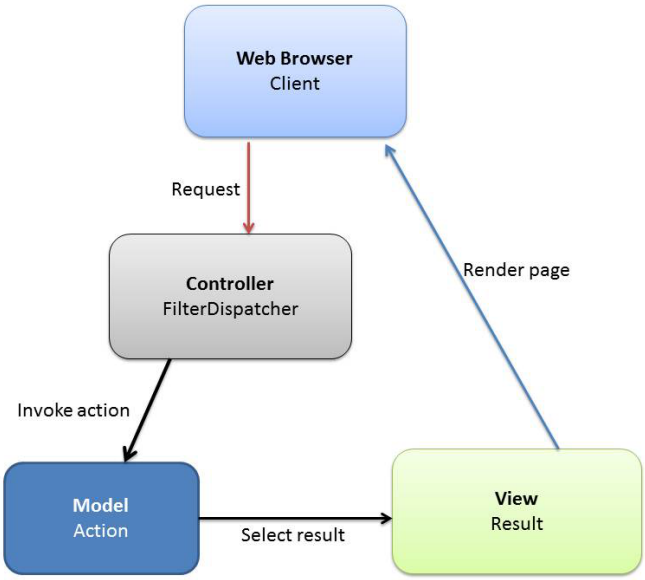
Předchůdci tohoto frameworku byli původně WebWork2 a Struts, kteří se v roce 2006 spojili do projektu Apache *Struts*^{2}. *Struts*^{2} je založen na tzv. MVC architektuře. Hlavními výhodami jsou:

- podpora jednoduchých anotací
- akční třídy jsou tvořeny POJO objekty
- podpora AJAX[2], DOJO[3]
- validace formulářů

5.5.1 Architektura *Struts*^{2}

Na obrázku 5.4 je znázorněno, jak framework *Struts*^{2} využívá architekturu MVC. Zpracování probíhá, tak, že

1. na klientském počítači je vytvořen request (požadavek), který nejprve projde předzpracováním přes filtry (SiteMesh, atd.)
2. následně ho obslouží *FilterDispatcher*, který vyvolá příslušnou akci
3. tato akce se provede v části *model*
4. z *modelu* je obsah předán do vrstvy *view*, která má za úkol vygenerovat výsledek provedené akce a poslat ho zpět klientovi



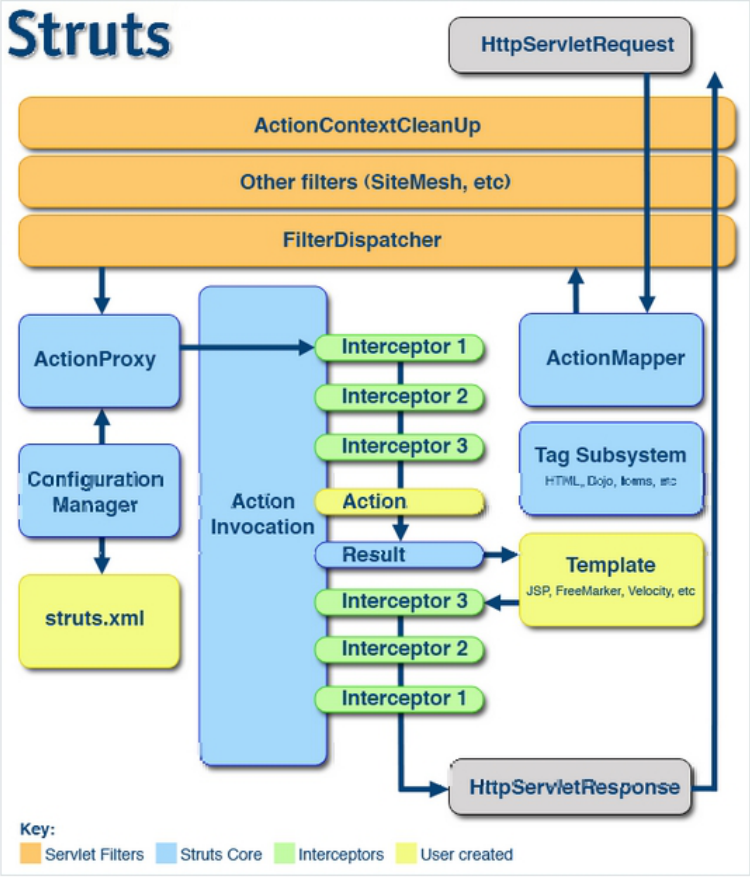
Obrázek 5.4: Tok dat *Struts*^{2} [22]

Kompletní architektura frameworku *Struts*^{2} je uvedena na obrázku 5.5. Je tvořena pomocí následujících komponent:

[2]Technologie vývoje interaktivních webových aplikací, které mění obsah svých stránek bez nutnosti jejich znovunačítání.

[3]Kolekcí javascriptových komponent, které mají pomoci webovému vývojáři.

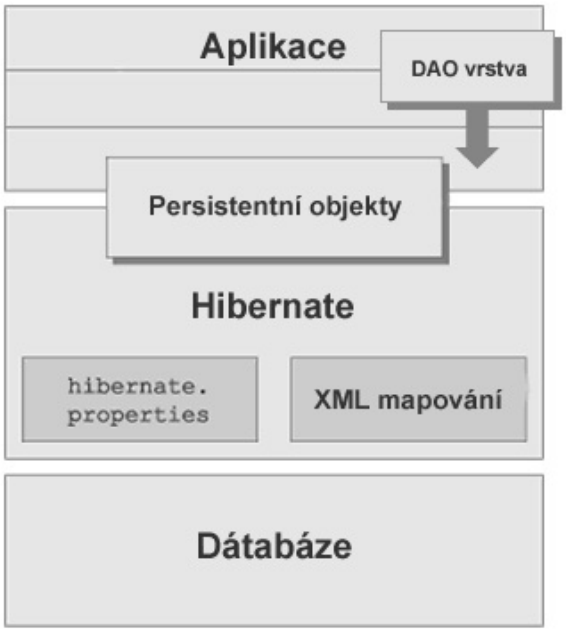
- ActionContextCleanUp - úvodní filtr, který provádí analýzu požadavku a následné ořezání daného kontextu
- FilterDispatcher - je součástí úvodních filtrů. Má za úkol přidělovat akce na základě requestu.
- ActionProxy - přečte nastavení ze souboru *struts.xml*
- Result - slouží k předávání odpovědí do vrstvy *view*
- Interceptor - slouží k zaobalení akce. Umožňuje přidání např. autentizace nebo formuláře pro upload souborů do aplikace Interceptory jsou vyvolány před a po provedení akce
- ActionInvocation - zodpovídá za implementaci návrhového vzoru command pattern
- ActionMapper - poskytuje bloku *FilterDispatcher* informaci o tom, zda je možné vyvolat akci



Obrázek 5.5: Architektura *Struts*^{2} [21]

5.6 Hibernate

Framework Hibernate se stal vzorem pro standardizaci objektově relačního mapování v Javě. Tento standard se nazývá JPA a od roku 2010 je framework Hibernate implementací tohoto standardu. Hibernate je knihovna pro objektově relační mapování v Javě. Objektově relační mapování se používá z následujícího důvodu - každá aplikace potřebuje někde uchovávat data. V současné době se k tomu obvykle používají relační databáze (například MySQL nebo Oracle).



Obrázek 5.6: Struktura frameworku Hibernate [8]

V relační databázi se nacházejí tabulky se sloupci a mezi tabulkami jsou prostřednictvím primárních a cizích klíčů vazby. Při programování v Javě se ale programuje objektově (třídy, atributy, vazby, dědičnost). V praxi je poté nutné transformovat data z relačního do objektového světa (a naopak). Právě ke zjednodušení a automatizaci této činnosti vznikl framework Hibernate. Framework Hibernate zajišťuje

- mapování na prostředí programovacího jazyka Java,
- odstínění uživatele (i programátora) od konkrétní databáze,
- trvanlivost (perzistenci) datové vrstvy,
- mapování tabulek na třídy, databázové relace na množiny, databázové atributy na java atributy.

Hibernate poskytuje způsob, díky kterému je možné zachovat stav objektů mezi dvěma spuštěními aplikace. Říkáme tedy, že udržuje data persistentní. Dosahuje

toho pomocí ORM, což znamená, že mapuje javovské objekty na entity v relační databázi. K tomu používá tzv. mapovací soubory, ve kterých je popsáno, jakým způsobem se mají data z objektu transformovat do databáze a naopak, jakým způsobem se z databázových tabulek mají vytvořit objekty. Druhý způsob, jak mapovat objekty, je použít anotace místo mapovacích souborů. V Hibernate se tedy pracuje s business objekty, pouze pro každý atribut přidáte get/set metody a metody *hashCode()* a *equals()*. Používají se tzv. POJO (Plain Old Java Object). Poté, co jsou objekty uložené v databázi, může se na ně dotazovat jazykem HQL (Hibernate Query Language), který je odvozen z SQL a je mu tedy velice podobný. [3] Obecné schéma aplikace využívající framework Hibernate je uvedeno na obrázku 5.6.

5.7 Apache Tiles

Apache Tiles představuje způsob, jak snížit množství stále se opakujícího kódu v aplikaci a současně umožňuje oddělit obsah od vzhledu a to lépe než directiva include. Funguje na principu šablon. Šablona je JSP stránka, která užívá knihovnu JSP uživatelských tagů k popisu rozložení stránky. Šablona funguje jako definice toho, jak budou stránky aplikace vypadat, aniž by určovala jeho obsah. Obsah se do stránek šablon vkládá za běhu. Stejnou šablonu může používat jedna nebo více stránek. Smysl šablon spočívá v získání soudržného vzhledu aplikace, aniž by bylo nutné napevno programovat každou stránku. Je rozumné, aby většina stránek používala stejnou šablonu; nicméně často se stává, že některé stránky v aplikaci mají odlišný vzhled, a tudíž je potřeba použít více než jednu šablonu. [17]

5.8 Architektura MVC

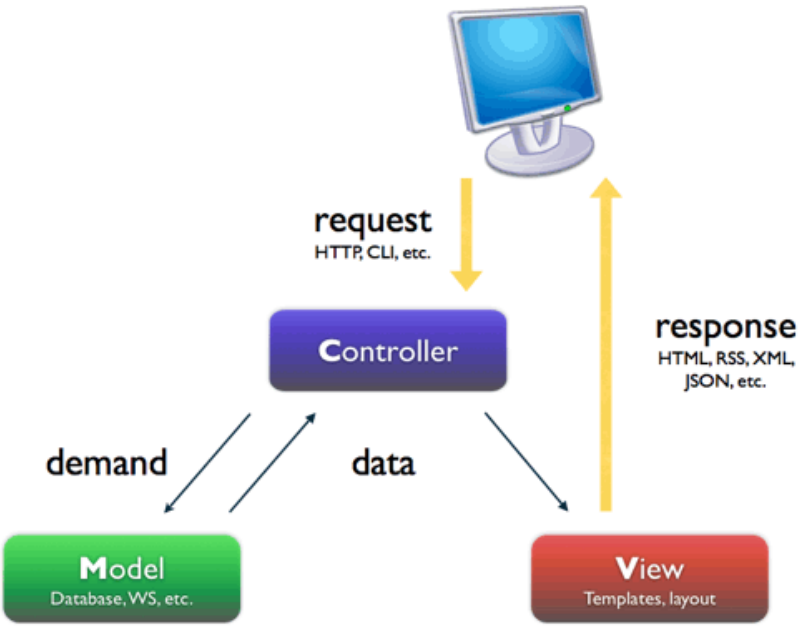
Jak již bylo dříve uvedeno, celá aplikace bude založena na architektuře MVC. Mezi největší výhody architektury MVC patří:

- jednoduchá správa aplikace
- snadná rozšiřitelnost a případná editace
- oddělení logiky od prezentace

Architektura MVC dělí aplikaci na tři logické části tak, aby je šlo upravovat samostatně a dopad změn byl na ostatní části co nejmenší. Tyto tři části jsou Model, View a Controller.

- model - reprezentuje data a business logiku aplikace
- view - zobrazuje uživatelské rozhraní
- controller - má na starosti tok událostí v aplikaci a obecně aplikační logiku

[15]



Obrázek 5.7: Architektura MVC

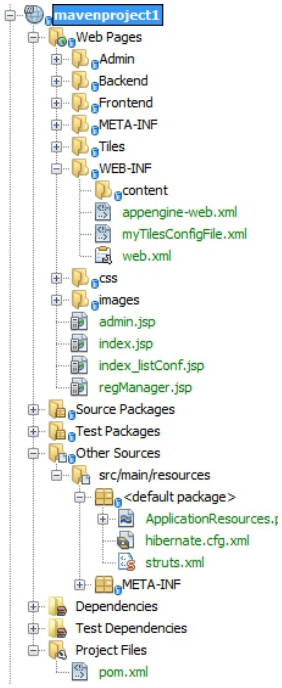
Na obrázku 5.7 je uvedena obecná interpretace architektury MVC.

6. Implementace

V této kapitole je popsána implementace CoMS nástroje ConfTUL podle návrhu z kapitoly 4.

6.1 Vytvoření a struktura projektu

K tvorbě aplikace bylo použito dobře známé vývojové prostředí NetBeans IDE. Prvním krokem je pochopitelně vytvoření nové aplikace. NetBeans má několik možností, jak webovou aplikaci vytvořit. Vzhledem k požadavku na využití nástroje Maven, byla vybrána možnost *Maven projekt*. Jeho výhodou, jak již bylo uvedeno, je to, že umožňuje jednoduché přidávání JAR souborů, které využívají pro svůj chod použité frameworky a také zjednodušení následného buildu aplikace. Struktura aplikace je uvedena na obrázku 6.1.



Obrázek 6.1: Adresářová struktura projektu

- Web Pages - obsahuje adresáře s JSP stránkami. *Admin* obsahuje stránky pro administraci aplikace, *Backend* stránky, kam se dostanou pouze registrova-

ní uživatelé a *Frontend* stránky, které jsou volně přístupné všem uživatelům. V adresáři Tiles jsou definovány templaty pro rozložení stránek a jednotlivá menu. Soubor *myTilesConfigFile.xml* uvádá rozložení stránek pomocí šablonovacího frameworku Tiles a *web.xml* základní nastavení frameworku *Struts*^{2}.

- Source Packages - obsahuje balíčky s java soubory
- Other Sources - obsahuje různá nastavení aplikace - soubor *hibernate.cfg.xml* pro nastavení frameworku Hibernate a soubor *struts.xml* pro nastavení frameworku *Struts*^{2}.
- Project Files obsahuje konfigurační soubor Mavenu *pom.xml*

6.2 Nastavení nástrojů

Prvním krokem bylo přidání odkazů do repozitáře do souboru *pom.xml*. Pomocí těchto odkazů byly staženy soubory, které potřebují jednotlivé frameworky *Struts*^{2}, Apache Tiles a Hibernate pro svůj běh. Seznam přidaných závislostí je uveden v příloze A.

6.2.1 struts.xml

Soubor struts.xml obsahuje nastavení frameworku *Struts*^{2}. Mimo jiné obsahuje

- nastavení akcí - obsahuje uživatelem vytvořené balíčky, které obsahují akce. Pomocí těchto akcí se provádí řízení celé aplikace.
- jazykovou lokalizaci
- validaci formulářů
- nastavení intereptorů

6.2.2 web.xml

Toto je konfigurační soubor pro J2EE, který určuje, jaké elementy requestu jsou zpracovávány pomocí servlet kontejneru. Je to vstupní bod pro všechny aplikace, také je v něm uvedena úvodní stránka. Nastavují s v něm dále například:

- doba platnosti session
- úvodní stránka
- parametry řídícího servletu
- mapování řídícího servletu

Soubor *web.xml* je uveden na obrázku 6.2

```xml
<?xml version="1.0" encoding="UTF-8"?>
<web-app id="struts_blank" version="2.4"
         xmlns="http://java.sun.com/xml/ns/j2ee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd">
    <display-name>Struts Blank Convention</display-name>
    <context-param>
        <param-name>org.apache.tiles.definition.DefinitionsFactory.DEFINITIONS_CONFIG</param-name>
        <param-value>/WEB-INF/myTilesConfigFile.xml</param-value>
    </context-param>
    <listener>
        <listener-class>org.apache.struts2.tiles.StrutsTilesListener</listener-class>
    </listener>
    <filter>
        <filter-name>struts2</filter-name>
        <filter-class>org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter</filter-class>
    </filter>
    <filter-mapping>
        <filter-name>struts2</filter-name>
        <url-pattern>/*</url-pattern>
    </filter-mapping>
    <welcome-file-list>
        <welcome-file>index.jsp</welcome-file>
    </welcome-file-list>
</web-app>
```

Obrázek 6.2: Konfigurační soubor web.xml

6.2.3 hibernate.cfg.xml

Soubor *hibernate.cfg.xml* slouží pro nastavení frameworku Hibernate. Je v něm potřeba nastavit ovladač na připojení k databázi, cestu k databázi, kódování, jméno a heslo a v neposlední řadě také namapovat všechny perzistetní třídy. Soubor je uveden na obrázku 6.3.

```xml
<?xml version='1.0' encoding='utf-8'?>
<!DOCTYPE hibernate-configuration PUBLIC
        "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
        "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">

<hibernate-configuration>
    <session-factory>
        <property name="connection.driver_class">com.mysql.jdbc.Driver</property>
        <property name="connection.url">jdbc:mysql://localhost:3306/conference</property>
        <property name="connection.useUnicode">true</property>
        <property name="connection.characterEncoding">UTF-8</property>
        <property name="connection.username">root</property>
        <property name="connection.password"></property>
        <property name="connection.pool_size">1</property>
        <property name="dialect">org.hibernate.dialect.MySQLDialect</property>
        <property name="current_session_context_class">thread</property>
        <property name="cache.provider_class">org.hibernate.cache.NoCacheProvider</property>
        <property name="show_sql">true</property>
        <property name="hbm2ddl.auto">update</property>

        <mapping class="cz.tul.dipl.model.User" />
        <mapping class="cz.tul.dipl.model.Address" />
        <mapping class="cz.tul.dipl.model.Country" />
        <mapping class="cz.tul.dipl.model.Conference" />
        <mapping class="cz.tul.dipl.model.Roles" />
        <mapping class="cz.tul.dipl.model.Paper" />
        <mapping class="cz.tul.dipl.model.Pages" />
```

Obrázek 6.3: Konfigurační soubor Hibernatu

6.3 Vizuální struktura aplikace

K tvorbě struktury aplikace je využit šablonovací framework Apache Tiles. Pomocí tohoto šablonovacímu systému je vytvořena ”kostra aplikace”, ve které jsou zobrazeny jednotlivé JSP stránky. Jak je uvedeno v části analýza, aplikace je rozdělena na tři systémy, přičemž bylo rozhodnuto, že budou vytvořeny dvě šablony, pomocí nichž budou tyto tři systémy zobrazeny.

První šablona bude složena ze tří částí (header, menu, body) a je využita pro frontend část aplikace a základ admin menu. Druhá šablona se skládá z levého a pravého menu a bude sloužit k administraci stránek. Finální vzhled vytvořených šablon je uveden na obrázku 6.4

Obrázek 6.4: Ukázka vzhledu pomocí šablon Tiles

Využití frameworku Tiles spočívá v následujících krocích:

1. pomocí výsledku provedené akce se uživatel dostane na JSP stránku

2. tato stránka musí obsahovat tiles definici, která je uvedena v souboru *myTilesConfigFile.xml*
3. pomocí této definice je vybrána šablona, ve které je definována templata a jednotlivé stránky

Příklad tiles definice je uveden na obrázku 6.5.

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@ taglib uri="http://tiles.apache.org/tags-tiles" prefix="tiles" %>
<tiles:insertDefinition name="aboutPage" />

<definition name="aboutPage" template="/Tiles/template.jsp">
    <put-attribute name="header" value="/Tiles/defaultMenu.jsp" />
    <put-attribute name="body" value="/Frontend/about_body.jsp" />
    <put-attribute name="footer" value="/Tiles/defaultHeader.jsp" />
</definition>
```

Obrázek 6.5: Nahoře: definice šablony. Dole: šablona.

6.4 Implementace architektury MVC

V sekci 5.7 byla vysvětlena architektura MVC. *Struts*^{2} také implementuje tento návrhový vzor a proto tvorba aplikace spočívá v implementaci jednotlivých vrstev.

6.4.1 Implementace modelu

Při objektově orientovaném programování je potřeba odpovědět na otázku, jak uložit objekt (černou skříňku, která obsahuje mnoho informací) do databáze, v níž jsou informace pouze atomické. Právě tento problém řeší tzv. POJO (Plain Old Java Object) třídy. Těm se také jinak říká třídy perzistentní.

K vytvoření perzistentních tříd je použit framework Hibernate. Použití frameworku Hibernate nevyžaduje žádné speciální rozhraní ani dědičnost. Vlastnosti perzistentních tříd:

- každá třída, která odpovídá jedné tabulce v databázi
- má shodné atributy, ke kterým se přistupuje metodami typu get a set
- třída dále obsahuje prázdný konstruktor
- jednoznačnou identifikaci je definován atribut Id typu Long

Takto vytvořené třídy se musejí namapovat do databáze. Mapování lze provést dvěma způsoby. Buď pomocí XML souborů (každé třídě náleží jeden XML soubor) nebo pomocí anotací obsažených přímo ve třídách. Framework *Struts*^{2} podporuje anotace, tudíž byla vybrána druhá z možností. Nastavení perzistentní třídy spočívá v:

```java
@Entity
@Table(name = "sponsor")
public class Sponsor implements Serializable {
    private byte[] image;
    private String nameSponsor;
    private String descriptionSponsor;
    private String linkSponsor;

    public Sponsor() {
    }

    @Id
    @Column(name = "NAME_SPONSOR")
    public String getNameSponsor() {
        return nameSponsor;
    }

    public void setNameSponsor(String nameSponsor) {
        this.nameSponsor = nameSponsor;
    }
```

Obrázek 6.6: Příklad perzistentní třídy

1. uvození třídy pomocí anotace *@Entity*
2. nastavení tabulky pomocí anotace *@Table*
3. implementace rozhraní *Serializable*, které pomáhá serializovat jednotlivé objekty
4. nastavení sloupců databáze pomocí anotace *@Column*

HQL, tvorba CRUD aplikace

Dotazovací jazyk Hibernatu (HQL) byl navržen jako minimální objektově orientované rozšíření základní obecné verze databázového dotazovacího jazyka SQL. HQL tím poskytuje elegantní přemostění mezi objekty a relačními architekturami databázových strojů. Přestože však HQL vychází ze syntaxe dotazovacího jazyka SQL, jedná se o plně objektově orientovaný jazyk, ve smyslu podpory dědění, polymorfismu nebo asociací. [16]

Při použití Hibernate a HQL je programátor tak trochu ”odstíněn” od zadávání dotazů do databáze. Zpravidla stačí načíst objekt z databáze a následnou práci s ním zajišťuje Java.

Zkratka CRUD je tvořena počátečními písmeny operací (Create, Read, Update, Delete), které by měla aplikace umožňovat. Samozřejmě existuje několik cest, jak tyto operace provádět. S využitím triviálních dotazů v HQL je možné realizovat takto:

Prvním krokem je vždy vytvoření objektu třídy SessionFactory. Díky objektu třídy SessionFacory je možné vytvářet sessions, pomocí kterých se získá spojení s databází. Pomocí transakcí (HQL dotazů) bude pak možné data v databázi upravovat.

Další operace jsou velice obdobné, jen je potřeba využít jednoduchý HQL dotaz. Opět je nutné vytvořit instanci třídy SessionFactory a začít transakci. Po vykonání

```
public Program add(Program program) {
    Session session = HibernateUtil.getSessionFactory().getCurrentSession();
    session.beginTransaction();
    session.save(program);
    session.getTransaction().commit();
    return program;
}
```

Obrázek 6.7: Příklad uložení pomocí SessionFactory

dotazu je transakce opět ukončená. Obdobným principem je možné provádět Read, Update i Delete. Příklad dotazu s využitím HQL je uveden na obrázku 6.8.

```
public Program delete(Integer idConf, String day, String time) {
    Program program;
    Session session = HibernateUtil.getSessionFactory().getCurrentSession();
    session.beginTransaction();
    Query q = session.createQuery("from Program where idConference = :idConf AND timeProgram = :time AND dateProgram = :day ");
    q.setParameter("idConf", idConf);
    q.setParameter("time", time);
    q.setParameter("day", day);
    program = (Program) q.list().get(0);
    session.delete(program);
    session.getTransaction().commit();
    return program;

}
```

Obrázek 6.8: Příklad mazání položky pomocí HQL dotazu

6.4.2 Implementace controlleru

Zjednodušeně se dá říci, že controller je část aplikace, která řídí spolupráci mezi datovou a zobrazovací částí aplikace. V praxi je controller *Struts*[2] aplikace tvořen pomocí FilterDispatcheru. Kromě úvodního filtrování je nejdůležitější činností FilterDispatcheru to, že na základě uživatelových requestů přiděluje, provádí a vyhodnocuje výsledky akcí. Tento proces probíhá v následujících krocích. Příklad je uveden na načtení programu pro určitý den:

1. Uživatel klikne na odkaz, který zobrazí program pro vybraný den konfenrece, tím je provedena akce *programDay* i s příslušnými parametry.

2. FilterDispatcher ověří v souboru *struts.xml*, zda je provést akce. K akci je (zpravidla) definována metoda, která se má provést, v tomto případě metoda *papersInProgram*.

3. Metoda se provede a na výsledku navrácené hodnoty se provede přesměrování na stránku, která je uvedená v souboru struts.xml.

Tento proces je zobrazen na obrázku 6.9.


```
<action name="programDay" class="cz.tul.dipl.view.ViewProgram" method="papersInProgram">
    <result name="success">Frontend/program.jsp</result>
</action>
```

```
public String papersInProgram() {
    HttpServletRequest request = ServletActionContext.getRequest();
    String getRequestId = request.getParameter("id");
    String getRequestIdDay = request.getParameter("idDay");

    for (int i = 0; i < controllerProgram.list().size(); i++) {
        if (controllerProgram.list().get(i).getIdConference().equals(Integer.parseInt(getRequestId)) &&
                controllerProgram.list().get(i).getDateProgram().equals(getRequestIdDay)) {
            listProgramPapers.add(controllerProgram.list().get(i));
        }
    }
    return SUCCESS;
}
```

Obrázek 6.9: Vyvolání akce, příklad akce, příklad metody, která je zavolána

6.4.3 Implementace vrstvy View

Vrstva view, neboli pohled, je část aplikace, která zodopovídá za zobrazování již zpracovaných informací. Tyto informace získá od controlleru, který je předtím zpracuje tak, aby mohly být zobrazeny touto vrstvou. K tomuto účelu jsou je využita kombinace JSP stránek a *Struts*^{2} tagů.

Tato vrstva by se neměla mít jinou funkci než zobrazování zpracovaných údajů. Častou chybou bývá, že se programátor snaží uplatňovat logiku pro zpracování dat i v této vrstvě, což je obtížné a vede ke znepřehlednění kódu. Výhodou je, že *Struts*^{2} podporuje jazyk OGNL[1].

Struts^2 tagy

Rozdíl mezi webovými aplikacemi a běžnými internetovými stránkami spočívá v tom, že webové aplikace mohou vytvořit dynamickou odezvu. Aby bylo snazší odkazovat se na dynamická data ze stránky, framework *Struts*^{2} nabízí sadu uživatelských značek. Některé z těchto znaků napodobují standardní HTML tagy a zároveň jim dávají přidanou hodnotu. Další tagy vytvářejí lehce nestandardní, ale zároveň užitečné ovládací prvky. [13].

Struts^2 tagy se dělí na dvě základní skupiny:

- Struts Generic Tags - používají se pro řízení toku dat při vykreslování stránky
  - Control Tags - používají se při řízení toku dat v tazích, jako jsou *IF, ELSE* nebo *iterátor*

    Na obrázku 6.10 je příklad využití tagu *s:iterator*. Tento tag se používá pro zobrazení všech prvků, které obsahuje struktura List nebo Map. Současně s ním je využit i tag *s:if*.

```
<%@ taglib prefix="s" uri="/struts-tags"%>

<h3>Členové organizačního výboru</h3>
    <s:iterator value="listRoles">
        <s:if test="orgCom == 1">
            <a href="mailto:${user.email}">
                <s:property value="user.name"/>
                <s:property value="user.surname"/>
            </a>
            <br/>
        </s:if>
    </s:iterator>
```

Obrázek 6.10: Příklad využití Struts Generic Tags

  - Data Tags - manipulace s daty, javaBeans, i18n

[1] Jazyk pro práci s objekty.

- UI Tags - určeny především pro data z akce nebo z data tagu. Tyto tagy se používají pro zobrazení na HTML stránce. UI tagy jsou řízeny tématy a šablonami

  – Form Tags - pro *Struts*^{2} formuláře

  Na obrázku 6.11 je využití tagu *s:form*, který obaluje *s:textfield, s:textarea, s:file* a *s:submit*. Pro vytvoření akce je využit *s:url*.

```
<s:url value='addPaper' var="urlAddpaper" escapeAmp="false">
    <s:param name="id" value="#parameters['id']"/>
    <s:param name="idPaper" value="#parameters['idPaper']"/>
</s:url>
<s:form action="%{urlAddpaper}" method="post" enctype="multipart/form-data" id="addPaper">
    <s:textfield name="paper.title" label="Název práce" size="80" id="title"
    <s:textarea label="Abstrakt" name="paper.abstrakt" cols="60" rows="23" id="abstract"
    <s:textfield name="paper.coWriter" label="Spolupracovníci" size="80" id="cowriter"
    <s:file name="file" label="Vyberte soubor" accept="application/pdf" onchange="checkSize(16777216,'submit')"
    <s:submit value="Odeslat" align="center" id="submit"/>
</s:form>
```

Obrázek 6.11: *Struts*^{2} formulář

  – Non-Form UI Tags - ostatní

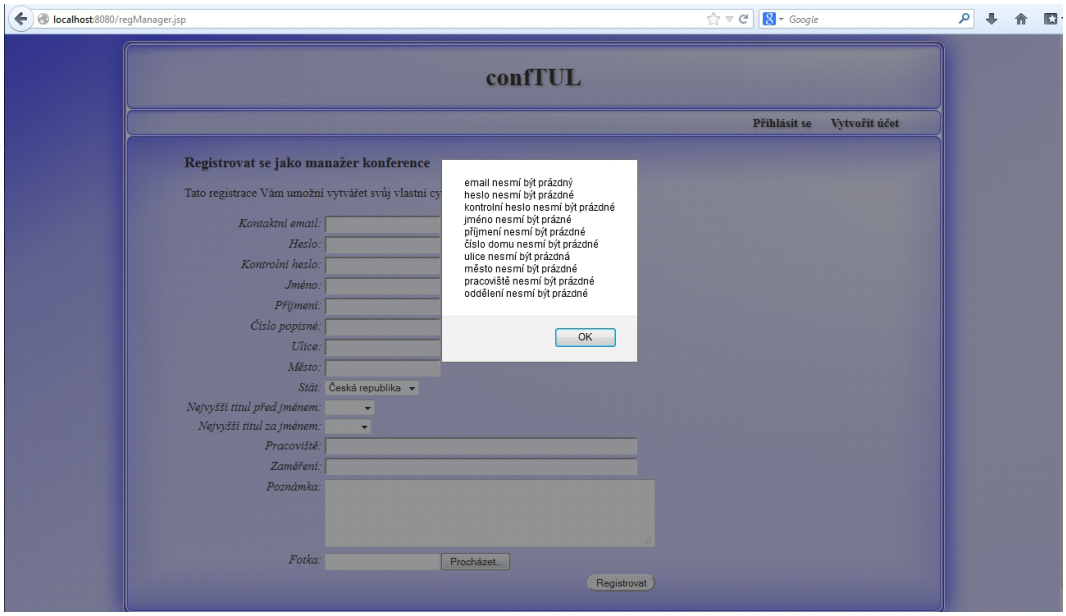
  – Ajax Tags - podpora Ajaxu

  Na obrázku 6.12 je uvedeno využití technologie AJAX, která je defaultně podporována pomocí *Struts*^{2} .

Obrázek 6.12: Využití kompomenty AJAX

Validace formulářů

Jak již bylo uvedeno v předchozích částech, jednou z výhod frameworku *Struts*^{2} je to, že obsahuje validaci formulářů, resp. dat, která se do nich vkládají. Problém ovšem nastává, pokud je validace použita tam, kde se využívá tzv. *action chaining*. *Struts*^{2} nepodporuje přenos validačních hlášení v případě, že je v akci chaining využit. V rámci zachování celistvosti aplikace bylo rozhodnuto, že bude využita validace formulářů pomocí JavaScriptu. Ukázka je uvedena na obrázku 6.13.



Obrázek 6.13: Validace pomoci JS

6.5 Nástroj ConfTUL

Podobně, jako byly v první kapitole otestovány funkce již hotových nástrojů na podporu konference, budou nyní vyzkoušeny i některé funkce nástroje ConfTUL.

6.5.1 Funkce nástroje

Nástroj confTUL by měl fungovat jako volně šiřitelný nástroj pro tvorbu konferencí. Uživatel, který se zaregistruje jako manažer konference, bude moci vytvářet a řídit svoje vlastní konference. Mezi vlastnosti nástroje confTUL patří:

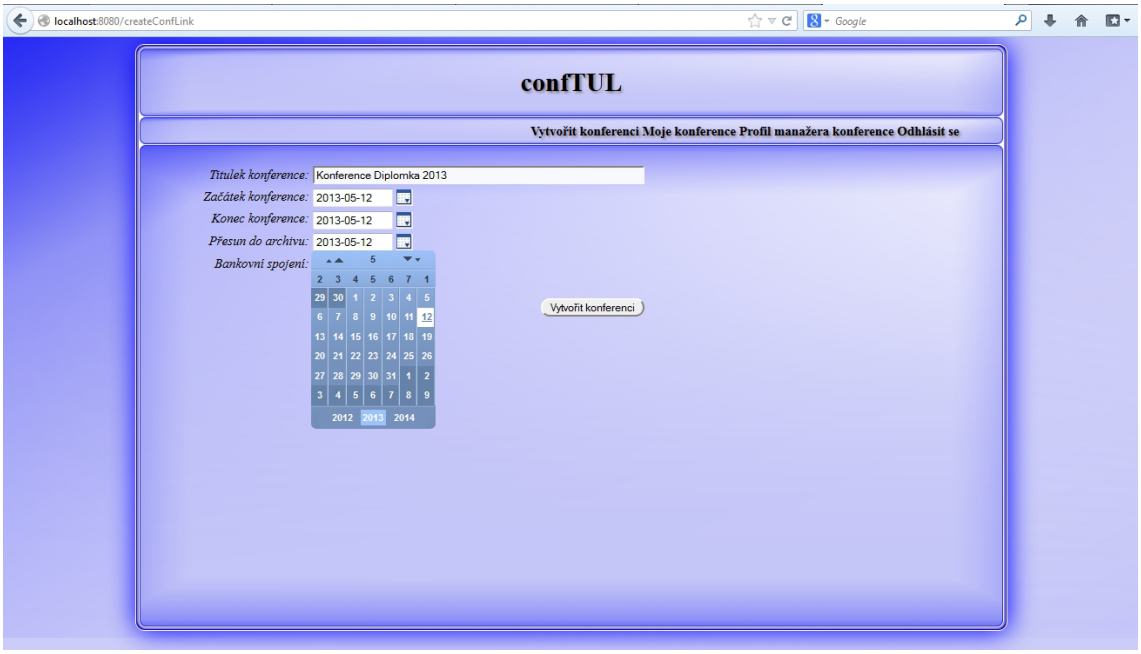
- tvorba libovolného počtu konferencí pro registrovaného uživatele
- uživatelské role se mohou vzájemně prolínat
- recenzní řízení
- generování programu podle přijatých příspěvků

6.5.2 Tvorba konference

Prvním krokem pro vytvoření konference je registrace uživatele jako manažera konference. Po registrování získá tento uživatel právo vytvářet svoje vlastní konference. Toto je možné v menu z obrázku 6.14.

Nejprve je nutné vytvořit konferenci. Tvorba konference je uvedena na obrázku 6.15. Pro vytvoření konference stačí zadat její titulek, časový harmonogram a bankovní spojení, pomocí kterého bude probíhat úhrada poplatků.

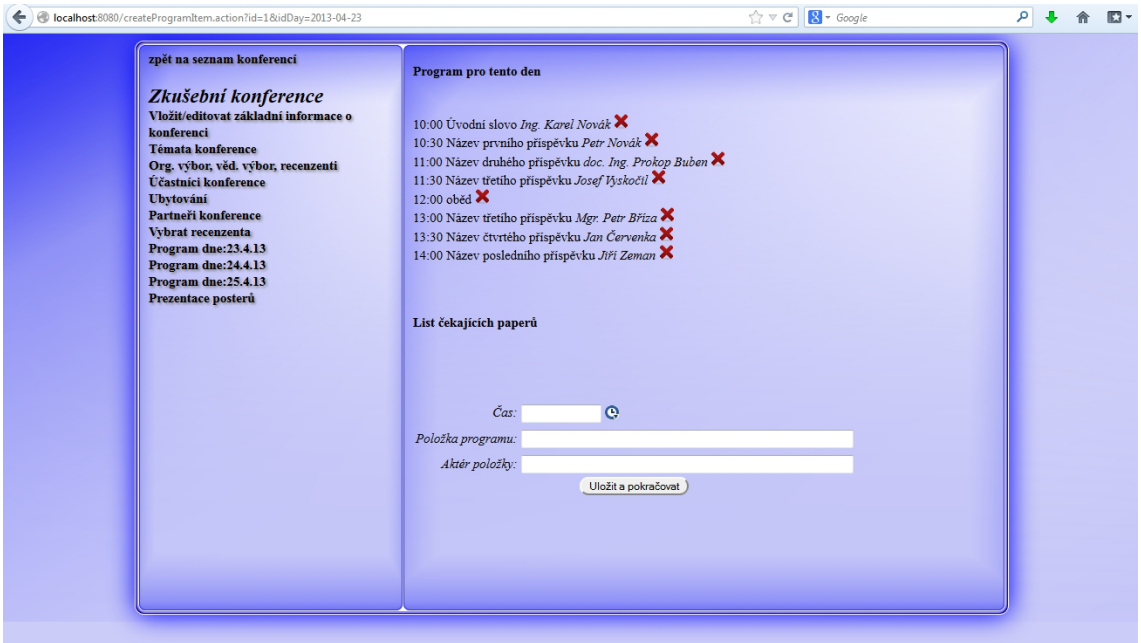
Obrázek 6.14: Manažerské menu



Obrázek 6.15: Tvorba konference

Byla vytvořena konference. Po jejím vytvoření musí její manažer nejprve zadat všechna potřebná data, která budou na stránkách zobrazena. To se provádí v menu z obrázku 6.16. Pomocí tohoto menu je de facto řízena celá konference. Její řízení lze rozdělit na několik částí.

- Vložit/editovat základní informace o konferenci. Pomocí této nabídky jsou vloženy a případně editovány základní informace o konferenci, mezi které patří:
  - informace uvedené na úvodní stránce konference
  - intrukce k abstraktu a příspěvku
  - detaily k placení konference
  - informace o pořadateli konference, kontakt
- *Témata konference*, *Ubytování*, *Partneři*
- Prezentace posterů - vložení informací týkajících se prezentace posterů
- Program - generování programu je uvedeno na obrázku 6.16. Dny programu jsou automaticky přidány podle trvání konference. Program se skládá z přijatých příspěvků a z ostatních položek. Přijaté příspěvky jsou zobrazeny v jejich seznamu, při kliknutí na příspěvek je do formuláře vložen jeho titulek a jméno jeho autora. Je přidán čas a příspěvek může být vložen do programu.



Obrázek 6.16: Generování programu

- Vybrat recenzenta - v tomto menu jsou zobrazeny všechny příspěvky konference, které procházejí recenzním řízením (např. příspěvky klíčových mluvčích jsou automaticky přijaty i bez recenzního řízení). Pro každý příspěvek

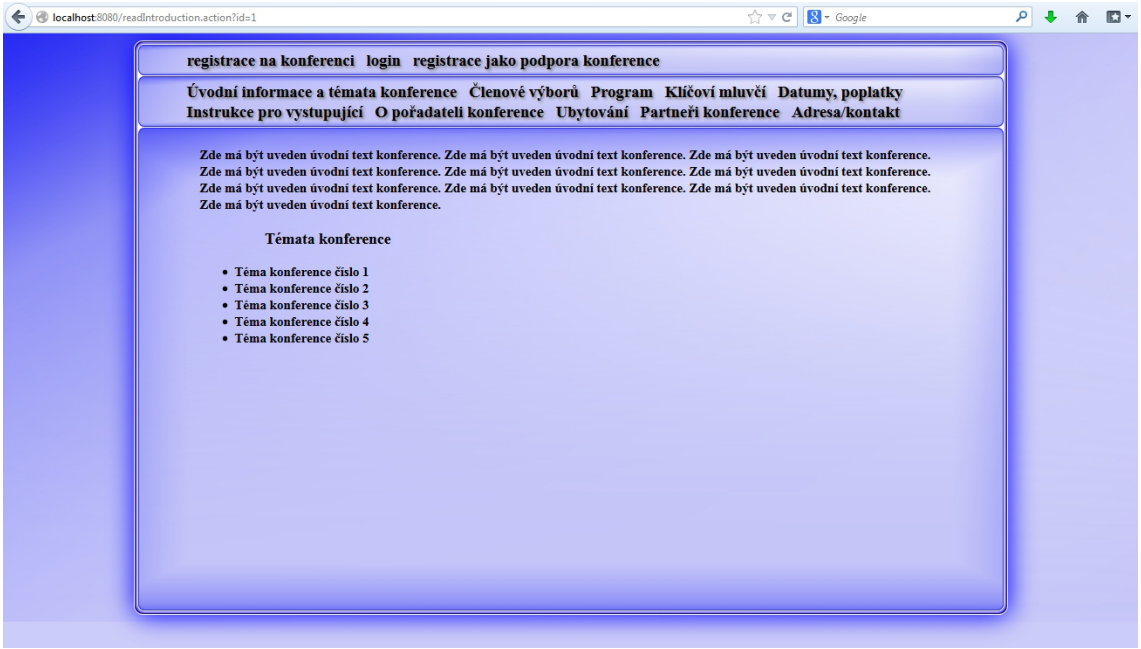
je vybrán jeden recenzent, který provede recenzní řízení a buď příspěvek na konferenci přijme nebo odmítne. Více v sekci *Recenzní řízení*

- Účastníci konference - seznam všech placených účastníků konference. V tomto menu je možné přepínání jejich rolí. Dále je zde řízena stupního účastnického poplatku.
- Org. výbor, věd. výbor, recenzenti - v tomto menu jsou řízeni řízeni všichni neplatící účastníci konference. Je zde opět možné přepíná rolí, odstraňování uživatelů, atd.
- Uveřejnění konference - posledním krokem po zadání všech údajů (které je ale možné kdykoliv editovat) je uveřejnění konference. Tím se konference zobrazí všem uživatelům.

6.5.3 Vytvořená konference

Vzhled vytvořené konference je na obrázku 6.17. Celkový vzhled je tvořen ze tří menu:

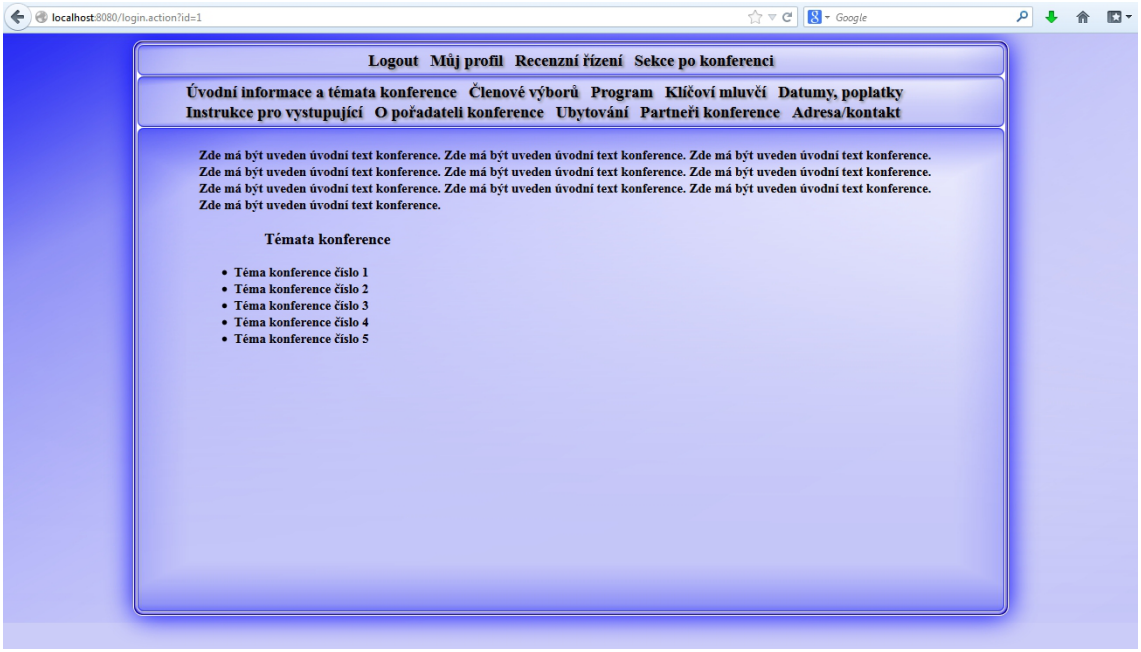
- část obsahující informace k vytvořené konferenci
- menu k části obsahující informace
- menu k přihlášení a registraci uživatelů



Obrázek 6.17: Úvodní stránka vytvořené konference

Aplikace umožňuje dva druhy registrací - jako podpora konference nebo jako návštěvník konference. Registrace je časově omezená datem, které stanoví manažer konference.

Po přihlášení se uživatel dostane do menu, které je uvedeno na obrázku 6.18. Zde je možné zobrazit všechny informace o konferenci, které byly při jejím vytvaření vloženy. Dále jsou zde dvě sekce, do kterých se dostane pouze registrovaný uživatel. Je to sekce *Recenzní řízení* a *Sekce po konferneci*. Ve druhé ze jmenovaných sekcí jsou uveřejněny všechny příspěvky, které byly na danou konferenci odeslány (i ty, které nebyly přijaty k prezentaci). Tato sekce je zpřístupněna až v době, kdy je konference převedena do stavu archivace (datum archivace vloží její manažer při tvorbě konference).



Obrázek 6.18: Přihlášený uživatel

6.5.4 Recenzní řízení

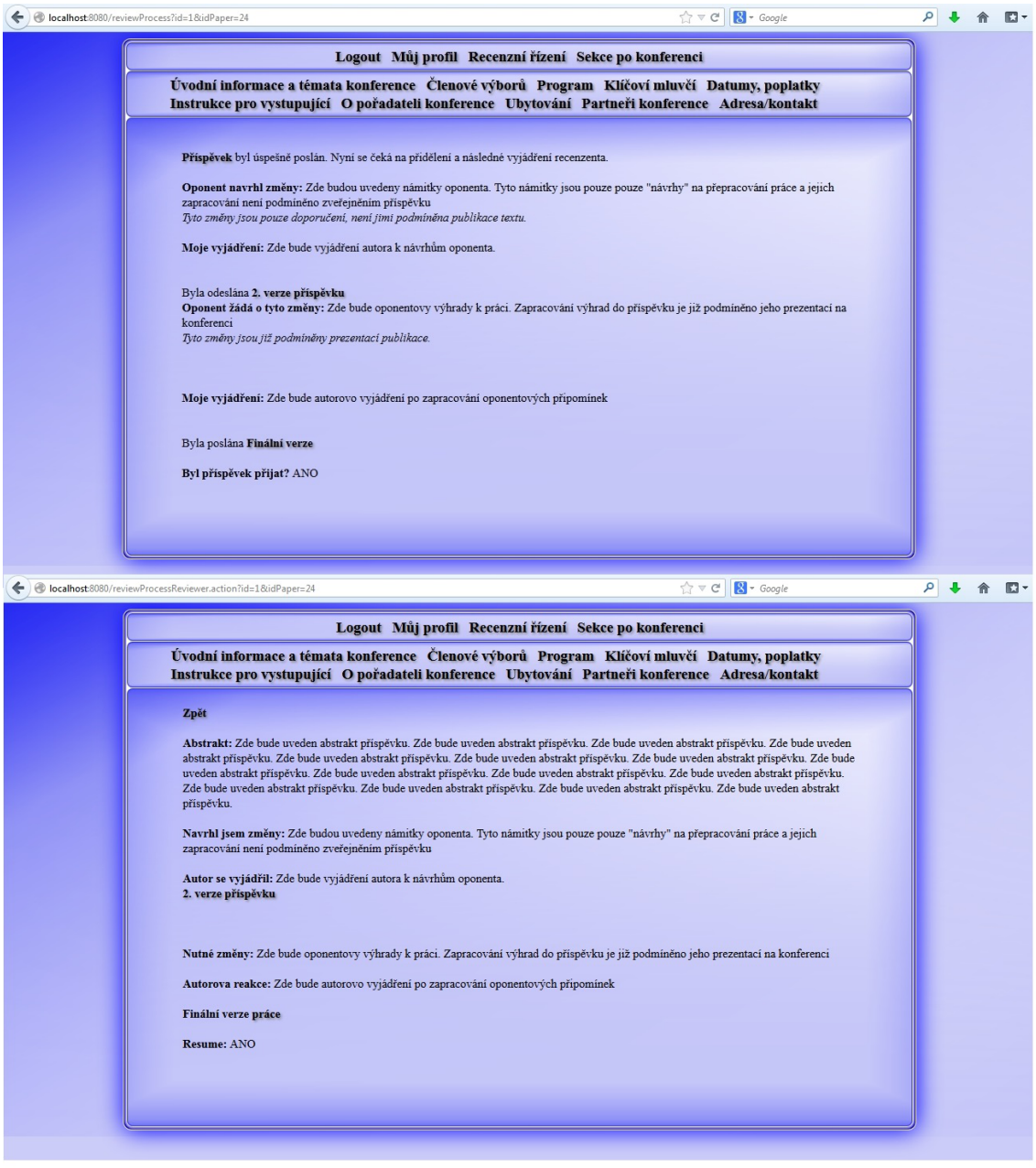
Sekce *Recenzní řízení* umožňuje, jak již její název napovídá, vstup do recenzního řízení konference. Toto se dějí na tři části podle typu uživatele:

- vstup do recenzního řízení není povolen - pro uživatele, kteří nemají příspěvek a současně nejsou recenzenty (např. posluchači konference nebo uživatelé s posterem)
- recenzní řízení umožňuje pouze zaslání příspěvku - pro klíčové mluvčí nebo pro uživatele, kteří si nepřejí příspěvek prezentovat
- recenzní řízení umožňuje recenzování příspěvků - pokud je uživatel v roli recenzenta a pokud je mu přidělen od manažera konference příspěvek k recenzování, je to prováděno v následujícím menu

Recenzní řízení je opět omezeno datumem, které stanoví manažer konference. Jak již bylo uvedeno v části o programu, přijaté příspěvky jsou nabídnuty ke vložení do programu v části o programu.

Recenzní řízení probíhá v několika krocích viz obrázek 6.19

1. uživatel zašle svůj příspěvek do recenzního řízení
2. manažer konference vybere vhodného recenzenta a přiřadí mu tento příspěvek
3. recenzent zobrazí příspěvek
    - recenzent přečte příspěvek a buď ho přijme nebo autorovi pošle návrh na změny, které ovšem nejsou podmíněny publikací
    - autor případné návrhy může zapracovat a pošle druhou verzi příspěvku
    - recenzent přečte příspěvek a buď ho přijme nebo autorovi pošle návrhy na změnu, tentokrát již podmíněné přijetím příspěvku
    - autor příspěvky zapracuje a pošle finální verzi příspěvku
    - recenzent buď příspěvek přijme nebo nepřijme



Obrázek 6.19: Recenzní řízení z pohledu uživatele a recenzenta

Závěr

Diplomová práce se zabývá webovými aplikacemi na podporu uspořádání konference. Prvním cílem bylo nalézt a otestovat nástroje, které k této činnosti slouží. Po otestování několika nástrojů bylo potřeba provézt jejich analýzu. Pomocí analýzy byl vytvořen návrh nové aplikace. Podle tohoto návrhu byla aplikace implementována.

Nalézt nástroje na podporu uspořádání konferencí není problém. Ovšem nalezení základních informací (technologie, platforma, detaily aplikace) není jednoduché z toho důvodu, že tyto aplikace jsou zpravidla vyvíjeny malými týmy vývojářů, kteří nedbají na podporu aplikace. Proto i samotná instalace nástroje může být často problém. Nicméně i přesto se podařilo některé nástroje nainstalovat a otestovat.

Dalším cílem bylo zjištění nevýhod univerzálních CMS nástrojů při jejich použití jako podpory při pořádání konference. Toto obsahuje druhá kapitola. Pomocí testování nainstalovaných nástrojů byl získán přehled toho, co by takové nástroje měly splňovat. Na základě vytvořeného přehledu byla vypracována UML analýza, která se použila pro implementaci nové aplikace. Avšak před samotnou implementací bylo třeba vybrat nástroje a technologie, pomocí kterých byla aplikace vytvořena. Závěr práce pojednává o implementaci jednotlivých vrstev MVC modelu, na kterém je aplikace vytvořena. V závěru práce jsou uvedeny příklady použití jednotlivých funkčností, které aplikace umožňuje.

Nově vytvořená aplikace obsahuje základní funkce, které by měl nástroj pro podporu odborné konference splňovat. Nicméně v porovnání s profesionálně vytvořenými aplikacemi je možné (a vhodné) další rozšíření aplikace tak, aby bylo pořádání konference ještě snazší. Vylepšením by bylo doplnit podporu některé platební brány (např. PayPal), možnost jazykového nastavení, aby mohla být aplikace užívána, globálně či větší modularitu při nastavování vzhledu aplikace.

Literatura

[1] Apache Tomcat. [online], Citováno dne 15. 5. 2013.
http://www.stahuj.centrum.cz/internet_a_site/servery/webove/apache-tomcat/.

[2] Conf2py - Conference Management System. [online], Citováno dne 15. 5. 2013.
http://www.findbestopensource.com/product/conf2py/.

[3] Hibernate. [online], Citováno dne 15. 5. 2013.
http://cs.wikipedia.org/wiki/Hibernate.

[4] Informace: Apache Maven. [online], Citováno dne 15. 5. 2013.
http://www.java-skoleni.cz/info/maven.php.

[5] Informační systém. [online], Citováno dne 15. 5. 2013.
http://cs.wikipedia.org/wiki/Informa%C4%8Dn%C3%AD_syst%C3%A9m.

[6] Joomla! [online], Citováno dne 15. 5. 2013.
http://cs.wikipedia.org/wiki/Joomla!

[7] MySQL. [online], Citováno dne 15. 5. 2013.
http://cs.wikipedia.org/wiki/MySQL.

[8] Objektově relační mapování: Hibernate. [online], Citováno dne 15. 5. 2013.
http://www.kiv.zcu.cz/~zima/vyuka/db2/cviceni-orm-01.html.

[9] Open Conference Systems. [online], Citováno dne 15. 5. 2013.
http://www.softaculous.com/apps/others/Open_Conference_Systems/.

[10] OpenConf. [online], Citováno dne 15. 5. 2013.
http://www.openconf.com/support/.

[11] Technologie Java EE. [online], Citováno dne 15. 5. 2013.
http://www.aura.cz/cz/produkty-a-sluzby/vyvoj-sw-na-zakazku/technologie-javaee/.

[12] Tomcat Overview. [online], Citováno dne 15. 5. 2013.
http://www.wellho.net/downloads/A651.pdf.

[13] Using *Struts*^{2} Tags. [online], Citováno dne 15. 5. 2013. http://struts.apache.org/release/2.1.x/docs/using-struts-2-tags.html.

[14] Základní funkce CMS. [online], Citováno dne 15. 5. 2013. http://www.manazer.bluefile.cz/?p=23.

[15] Bořek BERNARD. Úvod do architektury MVC. [online], Citováno dne 15. 5. 2013. http://www.zdrojak.cz/clanky/uvod-do-architektury-mvc/.

[16] Marek BĚHÁLEK. Hibernate. [online], Citováno dne 15. 5. 2013. http://www.cs.vsb.cz/behalek/frvs/2005/java/hibernate/hibernate.html.

[17] Chuck CAVANESS. *Programujeme Jakarta Struts*. Slavoj Písek, 2003.

[18] Well House Consultants. Tomcat Overview. [online], Citováno dne 15. 5. 2013. http://mordanky.blogspot.cz/2011/09/proc-pouzivat-framework.html.

[19] Miroslav MULLER Hana KANISOVÁ. *UML srozumitelně*. Computer Press, Brno, 2006.

[20] KVÍČALA LUDĚK. Luděk Kvíčala Webdesing. [online], Citováno dne 15. 5. 2013. http://www.ludekkvicala.cz/index.php?lnk=qrs.

[21] Viral Patel. Introduction to *Struts*^{2} Framework. [online], Citováno dne 15. 5. 2013. http://viralpatel.net/blogs/introduction-to-struts-2-framework/.

[22] Raghupathi PULISHETTI. *Struts*^{2} Framework . [online], Citováno dne 15. 5. 2013. http://javaguide4beginners.blogspot.cz/2013/01/struts2-framework.html.

A. Maven závislosti

- struts2-core
- struts2-convention-plugin
- struts2-junit-plugin
- struts2-config-browser-plugin
- struts2-tiles-plugin
- commons-logging
- junit
- servlet-api
- jsp-api
- commons-fileupload
- commons-io
- tiles-extras
- tiles-jsp
- tiles-api
- tiles-core
- slf4j-jdk14
- hibernate-core
- mysql-connector-java
- struts2-dojo-plugin
- struts-core
- javax.mail